

ANGLIA RUSKIN UNIVERSITY  
FACULTY OF SCIENCE AND ENGINEERING

**An Intelligent Decision-making Scheme in a Dynamic  
Multi-objective Environment using Deep  
Reinforcement Learning**

Md Mahmudul Hasan



A thesis in partial fulfilment of the requirements of  
Anglia Ruskin University for the degree of  
*Doctor of Philosophy*

## Acknowledgements

At first, I am grateful to the Almighty Allah who creates us and gave us the opportunity to live, think and enjoy this planet. After that, I would like to thank those people who immensely motivated me and was a great influence during my PhD study. My first and foremost gratitude goes to my supervisor Professor M Alamgir Hossain who endlessly supported and guided me to achieve this level. I am also thankful to my co-supervisor Khin Lwin for her suggestions, supports and motivation towards the optimisation field. I am enormously grateful to my co-supervisors Antesar Shabut and Maryam Imani for their prompt feedback, technical support and constructive criticism during my study. As for the people who have supported me throughout the journey and pushed me forward whenever I was about to lag behind include my lovely wife- Mahmuda, mother- Rajia Begum, father- Md Mazibar Rahman Sarker, and my three sisters. Without their unconditional love, affection, compassion and support, it would have never been possible. During this period, I have been blessed with two sons—Mihran and Safwan who gave me enormous sparkle to accomplish my study in due time. I also want to mention one of my mentors- Syed Akhter Hossain, who motivated me a lot to enrol in PhD. Undoubtedly, I am grateful to EU funded Erasmus Mundus Action 2 SmartLink project (Grant Agreement-20140858) for the scholarship. I am also grateful to the Department of Civil Engineering and the Built Environment of ARU where I collaborated in a project during my PhD and later, it became one of the test cases in this research. Apart from these, many other friends in Chelmsford and Bangladesh encouraged me to complete my study within due time and offered help in many critical situations. This research has utilised the data and findings from WQRGIS project (Frontiers of Engineering-SF1617\1\42, 2016-2017) that was funded by the Royal Academy of Engineering, UK. I also like to thank the Brazilian partners in WQRGIS project for their support and special thanks for helping us to translate some reports from Portuguese to English.

## Abstract

Real-life problems are dynamic and associated with a decision-making process with multiple options. We need to do optimisation to solve some of these dynamic decision-making problems. These problems are challenging to solve when we need trade-off between multiple parameters in a decision-making process, especially in a dynamic environment. However, with the help of artificial intelligence (AI), we may solve these problems effectively. This research aims to investigate the development of an intelligent decision-making scheme for a dynamic multi-objective environment using deep reinforcement learning (DRL) algorithm. This includes developing a benchmark in the area of dynamic multi-objective optimisation in reinforcement learning (RL) settings, which stimulated the development of an improved testbed using the conventional deep-sea treasure (DST) benchmark. The proposed testbed is created based on changing the optimal Pareto front (PF) and Pareto set (PS). To the best of my knowledge, this is the first dynamic multi-objective testbed for RL settings. Moreover, a framework is proposed to handle multi-objective in a dynamic environment that fundamentally maintains an equilibrium between different objectives to provide a compromised solution that is closed to the true PF. To proof the concept, the proposed model has been implemented in a real-world scenario to predict the vulnerable zones based on the water quality resilience in São Paulo, Brazil.

The proposed algorithm namely parity-Q deep Q network (PQDQN) is successfully implemented and tested where the agent outperforms in terms of achieving the goal (i.e. obtained rewards). Though, the agent requires higher elapsed time (i.e. the number of steps) to be trained compared to the multi-objective Monte Carlo tree search (MO-MCTS) agent in a particular event, its accuracy in finding the Pareto optimum solutions is significantly enhanced compared to the multi-policy DQN (MPDQN) and multi-Pareto Q learning (MPQ) algorithms.

The outcome reveals that the proposed algorithm can find the optimum solution in a dynamic environment. It allows a new objective to accommodate without any retraining and behaviour tuning of the agent. It also governs the policy that needs to be selected. As far as the dynamic DST testbed is concerned, it will provide the researchers with a new dimension to conduct their research and enable them to test their algorithms in solving problems that are dynamic in nature.

**Keywords:** Deep reinforcement learning, multi-policy, multi-objective optimisation, dynamic environment, deep Q network, vector rewards, benchmarks, water quality evaluation, resilience.

# Table of Contents

Acknowledgements .....	i
Abstract.....	ii
Table of Contents.....	iii
List of Figures.....	vii
List of Tables .....	x
Acronyms.....	xi
Symbols .....	xii
Chapter 1.....	1
Introduction .....	1
1.1    Motivation .....	4
1.2    Aims and Objectives .....	5
1.3    Research Questions .....	6
1.4    Main Scientific Contributions .....	6
1.5    Test Cases.....	7
1.5.1    Test case 1 .....	7
1.5.2    Test case 2 .....	8
1.6    Deliverables.....	10
1.7    Terminologies and Notes on Style .....	11
1.8    Organisation of the Thesis.....	13
Chapter 2.....	15
Literature Review .....	15
2.1    Introduction .....	15
2.2    Background .....	16
2.3    Intelligent Decision-making Process.....	17
2.3.1    Decision support system .....	20
2.3.2    Reviewing existing data-driven decision support systems .....	21
2.4    Dynamic Multi-objective Optimisation.....	22
2.4.1    Applications of dynamic multi-objective optimisation problems.....	24
2.4.2    Challenges of dynamic multi-objective optimisation .....	25
2.5    Reinforcement Learning.....	26
2.5.1    Multi-objective reinforcement learning (MORL).....	30

2.5.2	Reviewing existing approaches of MORL .....	32
2.6	Markov Decision Process (MDP) .....	35
2.6.1	Single objective Markov decision process .....	36
2.6.2	Multi-objective Markov decision process (MOMDP).....	36
2.7	Deep Reinforcement Learning (DRL).....	38
2.7.1	Basic architecture of deep reinforcement learning .....	40
2.7.2	Challenges in deep reinforcement learning .....	42
2.8	A General Framework for Deep Q Network (DQN).....	43
2.8.1	Q learning .....	43
2.8.2	Basics of a deep Q network (DQN).....	44
2.8.3	Q-function modifications in DQN .....	45
2.8.4	Prioritised experience replay for DQN .....	47
2.8.5	Reviewing different deep Q networks (DQNs).....	48
2.9	Optimisation Techniques for DRL .....	52
2.9.1	Gradient descent .....	53
2.9.2	Stochastic gradient descent (SGD) optimiser .....	54
2.9.3	Adam optimiser .....	55
2.10	Benchmarks for Multi-objective Optimisation.....	55
2.10.1	Reviewing existing benchmarks for MORL.....	55
2.10.2	Reviewing existing benchmarks for evolutionary approaches .....	58
2.10.3	Generating benchmark for the dynamic MORL .....	59
2.11	Reviewing the Metrics for Performance Evaluation .....	61
2.11.1	Reviewing performance metrics for MOO problems .....	61
2.11.2	Reviewing performance metrics used in DMO algorithms .....	63
2.12	Existing Process of Water Quality Evaluation in São Paulo, Brazil .....	65
2.13	Reviewing Machine Learning Studies for Water Quality Evaluation.....	68
2.14	Justification of the Study .....	71
2.15	Summary .....	73
Chapter 3.....		75
Methodology.....		75
3.1	Introduction .....	75
3.2	Research design.....	76

3.3	Approaches to defining the conceptual model .....	80
3.4	Method Details for Test Case 1 .....	82
3.4.1	Raw-image approach .....	83
3.4.2	Hardcode approach .....	84
3.5	Method Details for Test Case 2 .....	88
3.5.1	Data collection and preparation .....	91
3.5.2	Water quality parameter selection and resilience calculation .....	93
3.6	Summary .....	98
Chapter 4.....		101
Problem Settings and Experimental Setups .....		101
4.1	Introduction .....	101
4.2	Defining the Dynamic Multi-objective Optimisation Problem (DMOP) ..	102
4.3	Defining the Dynamics of the DMOP .....	103
4.4	Proposed Benchmark for a Dynamic Multi-objective Environment .....	105
4.5	Empirical Setups for Test Case 1 .....	108
4.5.1	The mathematical model for test case 1 .....	108
4.5.2	Experimental settings for test Case 1 .....	112
4.6	Empirical Setups for Test Case 2 .....	114
4.6.1	Mathematical model to formalise MOMDP for test Case 2 .....	114
4.6.2	Experimental settings for test case 2 .....	124
4.7	Summary .....	128
Chapter 5.....		130
Proposed Algorithm: Parity Q Deep Q Network (PQDQN) .....		130
5.1	Introduction .....	130
5.2	Deep Q Network (DQN) Selection .....	131
5.3	Policy Search in DQN .....	133
5.4	Meta-policy Selection Mechanism .....	136
5.5	Tracking the Optimal Policy .....	141
5.6	High-level Architecture of the Proposed Algorithm .....	148
5.7	Setup and Training the Model .....	155
5.8	Summary .....	156
Chapter 6.....		158

Results and Discussions.....	158
6.1    Introduction .....	158
6.2    Evaluating Criteria .....	159
6.2.1    Evaluating criteria for the proposed benchmark.....	159
6.2.2    Evaluation criteria for the considered environments .....	160
6.3    Performance Evaluation .....	162
6.3.1    Comparison of the existing and the proposed benchmark.....	163
6.3.2    Comparison of the elapsed training steps and earned rewards .....	165
6.3.3    Comparison of the performance of the different algorithms .....	172
6.3.4    Comparison of the true PF for identifying the vulnerable zones.....	176
6.3.5    Statistical Evaluation of the proposed algorithm.....	179
6.4    Strengths and Weaknesses of the Proposed Method .....	180
6.5    Limitations and Areas for Improvement .....	181
6.6    Summary .....	183
Chapter 7.....	185
Concluding Remarks and Future Directions.....	185
7.1    Final Remarks on the Benchmark .....	187
7.2    Final Remarks on Test Case 1 .....	189
7.3    Final Remarks on Test Case 2.....	191
7.4    Future Works.....	193
Ethical Considerations.....	197
References .....	198
Appendices .....	228
Appendix A: Treasure values and the Pareto frontiers .....	228
Appendix B: Sample WQR dataset .....	229
Appendix C: Keras implementation .....	230
Appendix D: Visualisation of the deep layers .....	231
Appendix E: Weight-bias distributions .....	237
Appendix F: Expert system and identified vulnerable zones .....	238

## List of Figures

Figure 1. 1: Deep-sea Treasure (DST) hunt environment as test case 1 .....	8
Figure 1. 2: A schematic view of test case 2.....	9
Figure 2. 1: Examples of intelligent applications.....	18
Figure 2. 2: Data-driven decision-making process .....	21
Figure 2. 3: A typical RL model .....	28
Figure 2. 4: A typical setup for multi-objective reinforcement learning environment ...	30
Figure 2. 5: Multi-objective optimisation in MORL.....	32
Figure 2. 6: Schematic diagram of the learning mechanism based on ANN .....	38
Figure 2. 7: Timeline for the evolution of deep reinforcement learning.....	39
Figure 2. 8: The system architecture of deep reinforcement learning.....	41
Figure 2. 9: Q value selection in a Deep Q network .....	46
Figure 2. 10: Connection of a prioritised experience replay memory in a DQN .....	47
Figure 2. 11: Optimization with gradient descent.....	54
Figure 2. 12: Existing benchmarks for MORL .....	57
Figure 2. 13: Targeted area (São Paulo) to implement the proposed framework .....	67
Figure 2. 14: Distribution of sampling points based on different types.....	67
Figure 2. 15: A flow diagram to select the necessary methods.....	74
Figure 3. 1: Working packages for this research work .....	78
Figure 3. 2: Overview of the system architecture .....	79
Figure 3. 3: An agent traversing within a single environment .....	82
Figure 3. 4: Traditional Deep-sea Treasure problem: frontier and reward distribution ..	83
Figure 3. 5: Raw image approach to solve the test case 1 (DST environments).....	84
Figure 3. 6: A simplified visualisation of the MOMDP.....	85
Figure 3. 7: Sub-criteria impact on each criterion .....	91
Figure 3. 8: Water resources of Brazil by Basin .....	93
Figure 3. 9: Threshold values for IVA and IQA based on the water quality .....	95
Figure 3. 10: Threshold values for IET on the water quality parameters in São Paulo ..	96
Figure 3. 11: Steps to form the RL agent in this study .....	98
Figure 3. 12: Process to repeat the work in a different dataset .....	99
Figure 4. 1: Two instances of dynamic deep-sea treasure (type-II).....	105
Figure 4. 2: Dynamic Deep-sea Treasure problem- Silver and Gold (Type III).....	106
Figure 4. 3: Dynamic Deep-sea Treasure problem- DST Attack by Enemy (Type IV) ..	107
Figure 4. 4: A conceptual model of the Markov Decision Process (MDP) of the simulated environment.....	108
Figure 4. 5: Visualisation of the deep layer for the dynamic DST environments.....	112
Figure 4. 6: Visualisation of the deep layer for test case 1 .....	113
Figure 4. 7: Distribution of triangular fuzzy number .....	114
Figure 4. 8: MDP for the resilient area selection .....	119
Figure 4. 9: High-level architecture of the RL implementation on the WQR dataset...	121



Figure 4. 10: A visualisation of knowledge gathering by the agent based on the resilient areas .....	123
Figure 4. 11: Structure of the DRL framework that is used in test case 2 .....	126
Figure 4. 12: Visualisation of the deep layer for test case 2 .....	127
Figure 4. 13: Parity Q deep Q network-based Policy selection .....	128
Figure 5. 1: Importance sampling based on the image in the Space Invaders game.....	133
Figure 5. 2: A visualisation of Q value mapping in DQN architecture for governing policies .....	137
Figure 5. 3: Object-relation mapping to find out the equilibrium between objectives in a PQDQN architecture.....	138
Figure 5. 4: Meta-policy selection in the dynamic DST environment (silver only) .....	140
Figure 5. 5: The agent's traversing in the dynamic DST (silver and gold).....	141
Figure 5. 6: Objective space and the tracking of global optima.....	142
Figure 5. 7: Sample state transition between different states.....	143
Figure 5. 8: Visualisation of the changing time while the agent is traversing in the dynamic environments .....	145
Figure 5. 9: The agent's traversing (clashes) in the dynamic DST (attack by enemy) in different timestamps.....	145
Figure 5. 10: Changing Pareto frontier (a) at timestamp 1 to 2 (b) at timestamp 2 to 3	146
Figure 5. 11: Changing Pareto frontier (a) at 7 <sup>th</sup> timestamp and (b) at 19 <sup>th</sup> timestamp.	147
Figure 5. 12: A visualisation of the trajectory based on the meta-policy (a) the best-case scenario and (b) the worst-case scenario.....	148
Figure 5. 13: Sample of the Pareto dominance (a) decision space (b) objective space.	149
Figure 5. 14: High-level architecture of the proposed parity-q deep q network .....	152
Figure 5. 15: Visualisation of the importance sampling while traversing in the dynamic DST.....	154
Figure 5. 16: Block-diagram for updating the Q value .....	155
Figure 5. 17: Proposed PQDQN in the dynamic environment.....	157
Figure 6. 1: All DST environments in this study .....	163
Figure 6. 2: Heatmap of average visited states over 100 agents for dynamic DST (silver only) .....	166
Figure 6. 3: Learning accuracy over 1M steps for dynamic DST (silver and gold).....	167
Figure 6. 4: Mean squared error over 1M steps for dynamic DST (attack by enemy) .	167
Figure 6. 5: Bias and weight distributions on the learning rate of 1E-03 for the dynamic DST (silver and gold).....	168
Figure 6. 6: Weight distributions on the learning rate of 1E-03 with conv =1 and 2 for the dynamic DST (attack by enemy).....	168
Figure 6. 7: Bias and weights distributions on the learning rate of 1E-03 for predicting water quality resilience .....	169
Figure 6. 8: PCA visualisation (night mode enabled) for the dynamic DST (silver and gold) .....	170
Figure 6. 9: PCA visualisation for the dynamic DST (attack by enemy).....	171
Figure 6. 10: t-SNE visualisation for the objectives in the WQR environment.....	171

Figure 6. 11: Performance comparison of the algorithms in dynamic DST (silver and gold) in terms of GD, IGD and HV. ....	173
Figure 6. 12: Performance comparison of the algorithms in dynamic DST (attack by enemy) in terms of GD, IGD and HV. ....	174
Figure 6. 13: Performance comparisons of the algorithms for WQR environment in terms of GD, IGD and HV. ....	175
Figure 6. 14: Efficient frontiers; red dots: best-known Pareto front and grey dots: Obtained Pareto frontier by PQDQN. ....	177
Figure 6. 15: Vulnerability in various zones. ....	177
Figure 6. 16: Vulnerable zones identified by the Parity-Q-Deep Q network based on IQA, IET and IVA. ....	178
Figure 6. 17: a) CETESB (2016)'s IQA mapping for IQA, IVA and IET. ....	178
 Figure D. 1: Bird's eye view of the deep layer network. ....	231
Figure D. 2: Visualisation of deep layer 1. ....	232
Figure D. 3: Visualisation of deep layer 2. ....	232
Figure D. 4: Visualisation of deep layer 3. ....	233
Figure D. 5: Visualisation of deep layer 4. ....	233
Figure D. 6: Visualisation of the output layer. ....	234
Figure D. 7: Visualisation of the stochastic gradient descent (SGD) layer. ....	234
Figure D. 8: Visualisation of the iterations. ....	234
Figure D. 9: Visualisation of the momentum layer. ....	235
Figure D. 10: Visualisation of the loss layer. ....	235
Figure D. 11: Visualisation of the decay layer. ....	235
Figure D. 12: Visualisation of the kernel layer. ....	236
Figure D. 13: Visualisation of the bias layer. ....	236
Figure D. 14: Visualisation of the random_uniform layer. ....	236
 Figure E. 1: Weight-bias distribution for dynamic DST (silver and gold). ....	237
Figure E. 2: Weight-bias distribution for dynamic DST (attack by enemy). ....	237
Figure E. 3: Weight-bias distribution for WQR environment. ....	237
 Figure F. 1: Login window of the ES. ....	238
Figure F. 2: ES to predict water quality resilience based IQA, IET and IVA. ....	239
Figure F. 3: Identified vulnerable stations of Zone 5 by the ES. ....	239
Figure F. 4: Identified the most vulnerable stations based on IQA. ....	240
Figure F. 5: Identified the most vulnerable stations based on IET. ....	240
Figure F. 6: Identified the most vulnerable stations based on IVA. ....	241

## List of Tables

Table 2. 1: Data-driven decision support systems .....	22
Table 2. 2: Categories of the machine learning (ML) techniques.....	27
Table 2. 3: Different MORL approaches .....	33
Table 2. 4: Reviewing policy optimisation and actor-critic approach .....	48
Table 2. 5: Reviewing different DQNs .....	49
Table 2. 6: Performance metrics and their uses in analysing DMO algorithms.....	63
Table 2. 7: Performance metrics and their use in analysing DMO algorithms for the statistical tests .....	64
Table 2. 8: Freshwater monitoring networks in São Paulo State - 2017 .....	66
Table 3. 1: Multi-criteria decision matrix .....	90
Table 3. 2: The quality variables of the Basic Network (freshwater) .....	94
Table 4. 1: Dynamic MOP environment types .....	104
Table 4. 2: Hyperparameters for test case 1 .....	113
Table 4. 3: Global index of importance intensity of WQI by AHP and TFN .....	115
Table 4. 4: Hyperparameters for test case 2 .....	124
Table 5. 1: Comparison of the analysed algorithms.....	132
Table 5. 2: Progression of vector function estimation based on the Q values over time with the PQDQN.....	144
Table 6. 1: Comparisons of PF and PS between existing and proposed benchmark ....	164
Table 6. 2: Average number of steps and total expected return (in thousands) for MPQ, MO-MCTS and MPDQN.....	165
Table 6. 3: Average number of steps and total expected return (in thousands) for the proposed PQDQN .....	166
Table 6. 4: Frontier features for the vulnerable zones .....	176
Table 6. 5: Student's t-test results of different algorithms in different environments ..	179
Table A. 1: DST testbed treasure values.....	228
Table A. 2: Treasure values for the Pareto Frontier (Silver and Gold).....	228
Table A. 3: Treasure values for the Pareto Frontier (attack by enemy- scenario 1) .....	228
Table A. 4: Treasure values for the Pareto Frontier (attack by enemy- scenario 2) .....	229
Table B. 1: Sample raw dataset for the WQR.....	229
Table B. 2: The threshold level to determine the resilience of IQA, IVA and IET .....	230
Table F. 1: Physical location of the most vulnerable zones .....	241

## Acronyms

AI	Artificial Intelligence
ANN	Artificial Neural Network
CETESB	Companhia Ambiental do Estado de São Paulo
CS	Coverage Scope
DMO	Dynamic Multi-objective
DMOP	Dynamic Multi-objective Optimisation Problem
DL	Deep Learning
DRL	Deep Reinforcement Learning
DST	Deep Sea Treasure
DQN	Deep Q Network
ES	Expert System
IET	Índice do Estado Trófico
IQA	Índice de Qualidade das Águas
IVA	Aquí Índice de Vida Aquática
MDP	Markov Decision Process
ML	Machine Learning
MPDQN	Multi Policy Deep Q Network
MPQ	Multi Pareto Q Learning
MO	Multi-objective
MOO	Multi-objective Optimisation
MOP	Multi-objective Problem
MOMCTS	Multi-Objective Monte Carlo Tree Search
MOMDP	Multi-objective Markov Decision Process
MOTSP	Multi-objective Travelling Salesman Problem
MS	Maximum Spread
OR	Operational Research
PCA	Principal Component Analysis
PF	Pareto Front
POF	Pareto Optimal Front
PS	Pareto Set
POS	Pareto Optimal Set
PL	Path Length
RL	Reinforcement Learning
RM	Replay Memory
SC	Success Ratio
SP	São Paulo
t-SNE	t-Distributed Stochastic Neighbor Embedding
VD	Variable Distance
WQ	Water Quality
WQI	Water Quality Index
WQR	Water Quality Resilience

## Symbols

$\gamma$	Discount factor
$\delta$	Global significance
$\alpha$	Learning rate
$\rho$	Optimism index
$\pi$	Policy
$\mu$	Probability distribution
$\lambda$	Significance of factors
$\sigma$	Significance of sub-factors
$\tau$	Time-specific parameter
$Q'(\theta')$	Target network
$\theta$	Updating network
$V^\pi$	Value function
$\vec{r}$	Vector reward

# Chapter 1

## Introduction

Today's human life is blessed with science and its various applications. Especially, artificial intelligence (AI) added a new dimension making people believed that human intelligence could be replaced artificially. However, intelligence itself is so massive, spontaneous, primitive, and uncertain that it may not be purely replicated or replaced in the near future. Nevertheless, there are strong scientific communities who believe in this replacement, and it is truly appreciated from an academic point of view (Jarrahi, 2018; King and Grudin, 2016).

However, the mechanisms of intelligence can be analysed in a certain boundary with building machine, agent, and system or even writing a computer program. This artificially developed system can assist human to take a better decision or act according to the set of rules defined by the human (Duan, Edwards and Dwivedi, 2019). In other words, the scientific community will have more success in developing the system which learns how to be intelligent and perform accordingly (Julian Togelius, 2007; Yannakakis and Togelius, 2015). This thesis focuses on building an intelligent decision-making scheme which deals with the dynamics in the multi-objective (MO) environment. More specifically, this research directs how to develop a computer application which learns to be intelligent and performs to identify the optimised solution in a dynamic multi-objective (DMO) environment using deep reinforcement learning (DRL).

Human life consists of various problems which are dynamic, multi-parameter based and complex. Each of them requires different steps to be followed to make a final decision, and it needs optimisation if more than one alternative is available. Therefore, multi-objective optimisation, a process to find an optimum solution for a problem, has become popular in recent days (Zaroliagis and Christos, 2005; Botte and Schöbel, 2019). Many problems involve continuous changing properties and need to find an optimal solution from many available solutions, which is very

challenging. For instance, booking a flight or hotel, arranging class routines to adapt to constant changes because of staff absence and room unavailability, deploying a military unit in a war and so on. These scenarios require dynamic optimisation since the decisions need to be changed very frequently depending on the situation. Another example can be the use of medicines for cancer patients where the target is not just to cure them within less time but also to minimise the side effects of the drugs (Preissner et al., 2012). The problem also entails the risks of any new conditions that may arise during medication.

A common way to solve these dynamic multi-objective optimisation problems (DMOPs) in the domain of computational intelligence is an evolutionary approach (Azzouz, Bechikh and Said, 2017; Lam, Branke and Abbass, 2005). However, recently, many pieces of scientific literature in multi-objective optimisation show a radically different perspective in solving the problems using multi-objective Markov decision process (MOMDP) especially using reinforcement learning (RL) techniques (Lizotte and Laber, 2016; Drugan et al., 2017; Bamakan, Nurgaliev and Qu, 2019). One of the major goals of this technique is to reach the set of solutions known as Pareto-optimal solutions (POS) which is as close as to the true Pareto-optimal front (POF). These techniques not only find the shape of the Pareto front but also help to investigate and decode interesting facts that the solutions might have (Gopakumar et al., 2018). In addition, recently Multi-objective Markov decision process (MOMDP) has received considerable attention not only for its applicability but also for solving practical multi-objective problems (Lizotte and Laber, 2016). To solve the MOMDP, the common approaches are to define the RL model using state, action and reward functions. The reward functions can be scalar or vector. However, according to the reward hypothesis (Sutton and Barto, 2018) the goal and purposes can be formalised with the maximisation of the expected value of the cumulative sum of a received scalar signal (i.e. reward). In other words, the resulting MOMDPs can always be transformed into a single objective MDPs with aggregated returns.

Nevertheless, Roijers et al., (2013) rejected Sutton's view questioning its application in real-world scenarios. They presented three static scenarios (i.e. known weights, unknown weights, and decision support scenario) where authors showed one or both conversions are impossible, infeasible or undesirable. Moreover, as far as DMOPs are concerned, very few studies have been conducted in this area due to the lack of testbeds (Azzouz, Bechikh and Said, 2017). In this study, this research gap has been addressed by proposing a dynamic multi-objective testbed (i.e. dynamic deep-sea treasure hunt) which may lead the researchers to do further investigation in this area. To the best of my knowledge, this is the first work in the context of dynamic multi-objective optimisation using DRL. Besides, an argument for the necessity of dynamic multi-objective optimisation benchmark for RL settings has been established since the complexity of the problem space and finding a solution is computationally intensive in a reasonable timeframe such as NP-hard or NP-complete problems (Plaisted, 1984). Moreover, an algorithm has been proposed which is primarily responsible to handle more than one objective in the defined dynamic environment. After that, an implementation of this algorithm has been considered to identify and predict the vulnerable zones based on water quality resilience in 22 zones in São Paulo (SP), Brazil, which ensures the applicability and efficiency of the proposed algorithm. This implementation breaks the boundary of theoretical knowledge and helps to solve a practical problem.

Regarding the implementation, the basic network has only been considered which has 461 data collection points. The flow measurements in water bodies are carried out by Companhia Ambiental do Estado de São Paulo (CETESB) in partnership with the Department of Water and Energy of the State of São Paulo. The results are obtained by measuring the flow in water bodies by reading scale to sampling the water. In 2017, the core network generated approximately 118,000 (e.g. physical, chemical, biological, bioanalytical and ecotoxicological) volume of data (Publicações e Relatórios | Águas Interiores, 2017). This implementation may also lead to solving some of the other dynamic real-world problems that we face every day.



## 1.1 Motivation

We live in an era where there is no question that technology has drastically changed the way we work. According to Dr Carl Frey and Dr Michael Osborne, the economists from Oxford University, 40% of all category of jobs are at risk of being lost due to automation (Benedikt Frey et al., 2013). It is inevitable that artificial intelligence (AI) and machine learning (ML) will have a serious influence on this replacement (Chris Graham, 2018) even in the policy-making (Federico Mor, 2018). There are two different schools of opinion regarding the impact of AI on humanity (Dwivedi et al., 2019; Zanzotto, 2019). One school believes that AI is very much likely to have a destructive impact on mankind (Clarke, 2019) whereas the other school expects AI to play a positive role in the progress of humanity (Woo, 2020). However, this debate can only be solved in future when AI technology will flourish to its fullest. In this automation process, there will be a significant impact on job sectors in future and AI will be the trailblazer of this digitisation (Syed et al., 2020).

To do so, the computational intelligence researchers will involve more into simulation using robots, augmented & virtual reality and gaming environment. In this whole process, games or gaming environment will be one of the key components to analyse different algorithms and simulate the problems and provide solutions. The obvious reason is that a gaming environment can act as a guinea pig to design, develop, implement, test, modify and improve the algorithms (Justin Francis, 2017). Following the same motto, this study addresses a gap in the domain of DMOP and propose a benchmark with the help of the simulated environments as a contribution to this field.

In this thesis, a dynamic gaming environment has been created where there are a set of conflicting objectives. As mentioned earlier, the objectives and constraints of the problems vary dynamically from each other and are always evolving. To solve this problem, evolutionary algorithms (EA) are widely used to deal with optimisation. However, due to the dynamics over time, DMOPs are more

challenging to be solved and EAs often face difficulties to solve them (Jiang et al., 2018).

In spite of that, there has been a growing interest (Arulkumaran et al., 2017) to solve the multi-objective optimisation in sequential decision making using RL especially deep reinforcement learning (DRL) after the success of DeepMind in 2015 (Mnih et al., 2015). This study is also motivated by that achievement and intends to add values in a deep RL perspective to solve the problem of dynamic multi-objective optimisation. In addition, a water quality test case has been taken into consideration which is encouraged by one of the very crucial human needs, especially in the 21<sup>st</sup> century. In this research, water quality resilience has been studied thoroughly and a machine learning (ML) technique (i.e. DRL) is used to determine the critical areas in one of the cities in Brazil. In this research, a novel method called parity Q deep Q network (PQDQN) has been proposed which is able to find the non-dominated solutions in the dynamic DST environment and predict the vulnerable zones based on the water quality resilience in a dynamic multi-objective environment. The agent interacts in these environments that are based on the multi-objective Markov decision process (MOMDP) and capable of obtaining rewards in the RL setting.

## **1.2 Aims and Objectives**

In this study, the primary aim is to address the challenges in the existing testbeds for dynamic multi-objective optimisation in the context of reinforcement learning. The secondary aim of this study is to investigate and develop an appropriate decision-making framework for a dynamic multi-objective environment.

To achieve these aims, the following objectives are identified:

- a) To investigate the current state-of-the-art of the dynamic multi-objective optimisation in the context of RL.
- b) To design and develop a conceptual and mathematical model for dynamic multi-objective optimisation in RL settings.

- c) To design and develop a new dynamic multi-objective optimisation testbed for RL settings.
- d) To design and develop a novel algorithm using deep reinforcement learning that can handle dynamics and optimise the decision in a multi-objective environment.
- e) To apply the proposed algorithm to solve a real-world problem which is identifying and predicting the vulnerable zones using water quality resilience in the state of São Paulo, Brazil.

### **1.3 Research Questions**

In this study the answers to the following research questions are investigated:

- Q1: Can the proposed benchmark address the gap in the DMOP research domain for RL settings?
- Q2: How a DRL based algorithm can handle multiple objectives and predict the vulnerable zones based on water quality?

### **1.4 Main Scientific Contributions**

The major scientific contributions of this research work are as follows:

- a. Design and development of a new and innovative testbed for dynamic multi-objective optimisation for RL settings.
- b. Objective-relation mapping (ORM) is used for the first time to construct a meta-policy (e.g., govern policy) among different objectives to find out the compromising solutions.
- c. A novel method has been developed to validate the real-world applicability of the proposed algorithm which identifies and predicts vulnerable zones based on water quality resilience in São Paulo, Brazil.
- d. Identifying the research gap through extensive literature review in the context of DMOP in RL settings.

## 1.5 Test Cases

### 1.5.1 Test case 1

Deep-sea Treasure (DST) is a gaming environment. It is a standard multi-objective problem as well as a testbed introduced by (Vamplew et al., 2011) for RL settings. This is one of the popular testbeds and has appeared in the literature many times in the context of multi-objective RL research. This environment consists of 10 rows and 9 columns with three different types of cells such as water cells where the vessel can traverse, sea ground cells that cannot be traversed as these cells are the edges of the grid and treasure cells that provide different rewards. DST game ends when the agent reaches the treasure cells.

Here, the agent controls a submarine that searches for treasures under the sea. The objectives of the agent are to find out the highest valued treasure within minimum time (i.e. conflicting way). It has got deterministic transitions with the non-convex frontier. The submarine starts from the top left corner of the grid and can move up, down, right and left. Unlike the single objective environment, the agent receives vector rewards. The rewards are consisting of a punishment of -1 (i.e. negative reward for RL) for every move and the value of achieved treasure which is 0 except the agent reaches the location of the treasure when the agent receives the treasure amount (i.e. positive reward for RL). The optimal Pareto front has 10 non-dominated solutions, one per each treasure in the grid. The front is globally concave with local concavities at the treasure value of 74, 24 and 8. The hypervolume value of the optimal front Pareto front is 10455. Figure 1.1 shows a classical and static DST testbed where the lowest treasure value is 1 and the highest is 124.

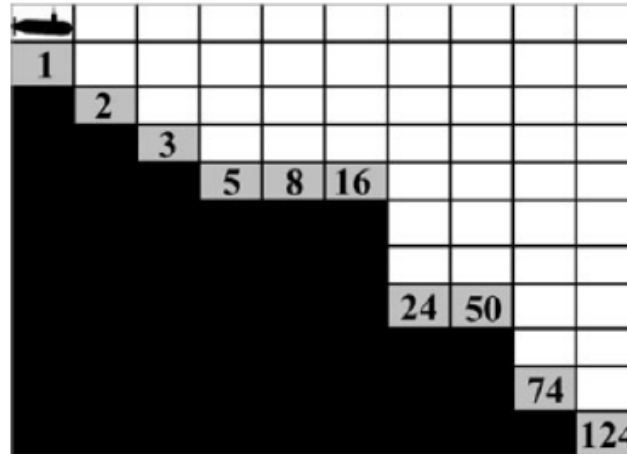


Figure 1. 1: Deep-sea Treasure (DST) hunt environment as test case 1

### 1.5.2 Test case 2

The test case 2 is selected so that the proposed algorithm (i.e. PQDQN) and the method (i.e. MOMDP) can solve a real-world problem in one of the crowded cities in Brazil. Considering the public water supply for this large population, the Government of the State of São Paulo is working for universal sanitation in municipals of the state where the increase in the percentage of the population served by various services such as measuring and maintaining the water quality, sewage services and so on. However, water contamination deteriorates the quality of water and impedes the sustainable development of São Paulo (Governo do Estado de São Paulo | Eleições, 2018). The presence of sewage in the waters of rivers, reservoirs, estuaries and coastal regions reduces water quality and restricts its multiple usages while increasing the occurrence of waterborne diseases caused by the primary contact or by the ingestion of contaminated water (Nogueira et al., 2018).

To identify the vulnerable areas and take proper actions in those areas, massive human efforts and expenses are required. These actions involve integrated management of actions involving various sectors and organisations associated with the management of the use of industrial and

agricultural effluents, the complexity of the human resource (HR) management, fixed asset and reactive or planned maintenance (Barbosa, Alam and Mushtaq, 2016). Therefore, it is important to automate the process to detect the vulnerable areas as quick as possible. Hence, an AI-based optimal decision support system can reduce the cost of managing such huge tasks and can have a socio-economic impact which may help for sustainable development. Figure 1.2 shows the bird's eye view of the test case 2 where the agent is capable to predict the vulnerable zones based on the water quality resilience.

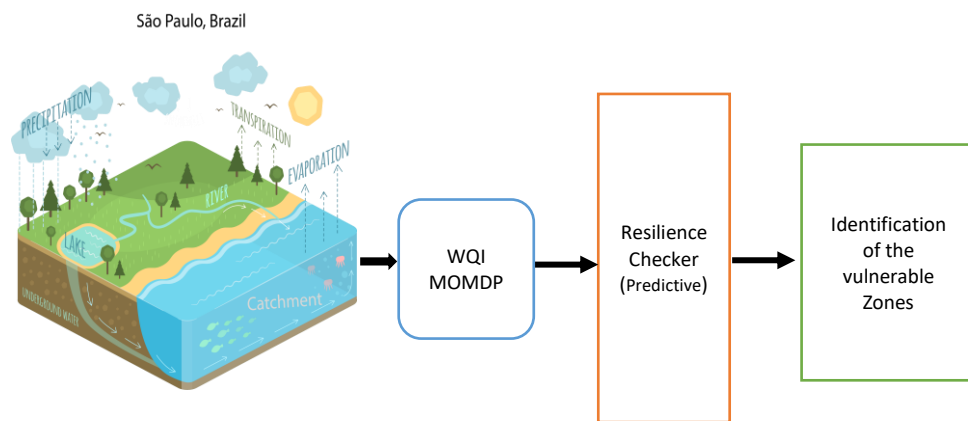


Figure 1. 2: A schematic view of test case 2

In a nutshell, the identified problems in this test-case are given below:

1. It is a dynamic problem considering the water quality data changes over time due to various factors.
2. Collection of these data is expensive and requires human resources.
3. Identification of the vulnerable zones is difficult because of manual checking and calculation.
4. Investment optimisation for different zones is complicated.
5. Prioritise the zones to enhance the water quality is time-consuming.

## 1.6 Deliverables

Deliverables in this research are given below as a list of publications.

### Journal:

**Md Mahmudul Hasan**, Khin Lwin, Maryam Imani, Antesar Shabut, Luiz Fernando Bittencourt, M.A. Hossain, “*Dynamic multi-objective optimisation using deep reinforcement learning: benchmark, algorithm and an application to identify vulnerable zones based on water quality*”, Engineering Applications of Artificial Intelligence, Publisher: Elsevier, Volume 86, 2019, Pages 107-135, ISSN 0952-1976, <https://doi.org/10.1016/j.engappai.2019.08.014>.

### IEEE Conferences:

1. **Md Mahmudul Hasan**, Khin Lwin, Antesar Shabut, Alamgir Hossain, “*Design and Development of a Benchmark for Dynamic Multi-objective Optimisation Problem in the Context of Deep Reinforcement Learning*”, 22<sup>nd</sup> International Conference on Computer and Information Technology, Dhaka, 2019. IEEE Xplore Digital Archive  
Link: <https://ieeexplore.ieee.org/document/9038529>
2. **Md Mahmudul Hasan**, Ali Mohsin, Maryam Imani, Luiz Fernando Bittencourt, “*A novel method to predict water quality resilience using deep reinforcement learning in São Paulo, Brazil*”, International Conference on Innovation in Engineering and Technology (ICIET), Dhaka, 2019.
3. **M. M. Hasan**, K. Abu-Hassan, Khin Lwin and M. A. Hossain, “*Reversible decision support system: Minimising cognitive dissonance in multi-criteria based complex system using fuzzy analytic hierarchy process*,” 2016 8th Computer Science and Electronic Engineering (CEECE), Colchester, UK, 2016, pp. 210-215. IEEE Xplore Digital Archive.  
Link: <https://ieeexplore.ieee.org/document/7835915>

### Other International Conference:

1. **Md Mahmudul Hasan**, Khin Lwin, Antesar Shabut, Miltu Kumar Ghosh, M A Hossain, “*Deep Reinforcement Learning for Dynamic Multi-objective Optimisation*”, 17<sup>th</sup> International Conference on Operational Research-KOI 2018, Zadar, Croatia, 2018.

### Other Contributions:

1. **Md Mahmudul Hasan**, Md Shahinur Rahman, Khin Lwin, Antesar Shabut, Adrian Bell, M A Hossain, “*Deep Reinforcement Learning for Optimisation*”, book chapter of ‘Handbook of Research on Deep Learning Innovations and Trends’, publisher: IGI Global, 2018.  
Link: <https://www.igi-global.com/chapter/deep-reinforcement-learning-for-optimization/227852>
2. **Technical Reviewer** for the book of “Machine Learning for Developers” Published by PACKT Publishers in 2017.  
Link: <https://www.packtpub.com/big-data-and-business-intelligence/machine-learning-developers>
3. **Md Mahmudul Hasan**, “*Predicting Water Quality Resilience: A Machine Learning Approach*”, 8<sup>th</sup> FST Conference, ARU, UK, 2019.
4. **Md Mahmudul Hasan**, “*A robust decision support system in dynamic multi-objective optimization using deep reinforcement learning*”, 12<sup>th</sup> Research Student Conference, ARU, UK, 2018.
5. **Best PhD Poster** publication at 7<sup>th</sup> FST Conference, ARU, UK, 2017.
6. **Md Mahmudul Hasan**, “Optimising decision in a multi-criteria based environment”, seminar at ARITI, ARU, UK, 2017.

### 1.7 Terminologies and Notes on Style

The following section represents common terminologies that have been frequently used in this study.

**Agent:** The agent or algorithm lives in the simulated environment and helps to make the decision.

**State:** A state helps to identify the next step which will be determined by the agent.

**Action:** Agent’s possible moves between different states by observing new state and receiving rewards.

**Policy:** A policy typically represents the agent’s behaviours of the selection of the action.



**Environment:** An environment is the external entity of the agent where it interacts with the states. The environment can be fully observable (i.e. the agent directly observes the environment) or partially observable (i.e. the agent indirectly observes the environment).

**Static environment:** The environment that does not change or being affected by the changing parameters and constraints.

**Dynamic environment:** The environment that changes over time. More specifically, the changing states influenced by the objective functions, constraints and problem parameters.

**Reward:** The agent has a specific task which needs to be performed by the actions. In a finite horizon or episodic environment, the expected return is usually an undiscounted finite sum of the scalar rewards until the agent reaches the terminal states.

**Decision space:** This term is used to define the space of the selections that represent the choices to make a decision.

**Objective space:** This space defines dominated and non-dominated solutions based on the objectives.

It is worth mentioning that a minimum usage of acronyms and mathematical terms have been used to read this thesis reasonably easy and enjoyable to the readers. At the end of some chapters, a graphical representation has been provided to give a visualisation and conceptual understanding. Moreover, there are some places where the mathematical equations are described in a readable format. However, readers have been referred adequately in certain places so that they can gather more information from relevant sources. In addition, some words (e.g. quick, slow, fast, long) have been used to exemplify the performance in the context of convergence, elapsed training time and identifying the true PF which have been widely stated

and utilised due to the approximation of the true PF (e.g. moving global optima) in the fields of optimisation and RL (Moffaert and Nowé, 2014; Lin et al., 2017; Farina, Deb and Amato, 2004; Mehnen, Wagner and Rudolph, 2006; Sutton and Barto, 2018). Furthermore, prior familiarity with reinforcement learning could have a notable impact on readers to follow and enjoy the reading.

## **1.8 Organisation of the Thesis**

The organisation of the thesis is illustrated below:

Chapter 2 reviews related research works where it highlights an overview of the intelligent application, decision support system, Markov decision process, machine learning, reinforcement, deep reinforcement learning, existing benchmarks and optimisation techniques. This chapter also represents a comprehensive analysis of the essential components to enhance the readability of the outcomes of the thesis such as reviewing performance metrics to analyse the algorithms. Finally, the justification of the study has been described in this chapter.

Chapter 3 deals with the methodology of the research where the research design is explained. This chapter also deals with the method details and the necessary approaches for doing this research. It has also provided a comprehensive analysis of the data-preparation, water quality parameter selection and the ways of the calculating resilience.

Problem settings and experimental setups are discussed in Chapter 4 where the mathematical and conceptual models have been described. In this chapter, the proposed benchmark, network architecture and a detailed discussion of formalising the MOMDP for the real-world scenario and the experimental setups for both of the test cases have been described.

The high-level architecture of the proposed algorithm has been explained in Chapter 5. In this chapter, a step by step working procedure has been explained for the proposed algorithm. In addition, this chapter also discusses the tools such as necessary software, libraries and machine configuration to develop the proposed algorithm.

Empirical analysis and discussions have been presented in Chapter 6 where the critical review and the limitations are also elaborated. In this chapter, the performance measuring criteria and the rationale behind choosing them are also mentioned. Furthermore, the strengths and weaknesses of the proposed algorithm have been explained.

Finally, concluding remarks and the future direction of this thesis have been stated in Chapter 7. The future direction includes an immediate and long-term goal for carrying out the existing research. This chapter also explains the further possible directions for both test cases.

## **Chapter 2**

### **Literature Review**

#### **2.1 Introduction**

This is a threefold study. The major part discusses the gap in the dynamic multi-objective optimisation in the context of reinforcement learning. After that, a benchmark has been created to address that gap and finally, an algorithm has been proposed that can effectively optimise multi-objective in a dynamic environment both in simulated and in a real-world scenario. In this section, the necessary components of this study and the recent works in this domain have been discussed. This chapter will help the readers to grasp the core elements and background knowledge in the following chapters.

At first, the intelligent decision-making scheme along with comparisons between the existing systems have been described. Then, one of the key components has been discussed in this research which is the dynamic environment and how it is related to the optimisation area. In this section, problems and the challenges of the dynamic multi-objective optimisation have been described broadly. After that, a comprehensive discussion has been conducted based on reinforcement learning and how it is going to solve single-objective and multi-objective problems. Then, one of the solving techniques of the RL will be discussed which is the Markov decision process (MDP). In this thesis, the MOMDP has been chosen to incorporate with RL settings. Therefore, this section will provide a substantial amount of the review of the existing solution to solve the problem of multi-objective RL problems. In addition, a basic component of the deep reinforcement learning and the major architecture of the deep Q networks (DQNs) have been explained. In this section, a comprehensive analysis and the advantages and disadvantages of the different DQNs will be discussing. After that, optimisation techniques and performance metrics to evaluate the algorithms will be conferring which have been used in this study. Finally, different machine learning approaches of the test case 2 will be discussed along with the justification of the study.

## 2.2 Background

Optimisation problems can be categorised into three broad areas, such as single objective, multi-objective, and many objectives problems (Types of Optimization Problems | NEOS, 2018). The other classifications of the optimisation problem are whether the problem is static or dynamic. In both cases, there is a good number of researches that have already been conducted using evolutionary approaches. For examples, 14 test problems for dynamic multi-objective optimisation called DF1 to DF14 were proposed (Jiang et al., 2018) based on PF/PS geometrics, irregular PF shapes, and disconnectivity. Multi-objective test problems with BIAS commonly known as BT1-BT9 (Li, Zhang and Deng, 2017) and Large-scale multi-objective test Problems (i.e. LSMOP1–LSMOP9) are proposed by (Cheng et al., 2017). F1–F10 for IM -MOEA by (Cheng et al., 2015), C1\_DTLZ1, C2\_DTLZ2, C3\_DTLZ4, IDTLZ1, IDTLZ2 (Deb and Jain, 2014) were proposed in 2014. In addition, a comprehensive list of the benchmarks for evolutionary optimisation can be obtained in (Tian et al., 2017; Li et al., 2008; Jiang et al., 2018).

On the other hand, in the domain of multi-objective reinforcement learning (MORL), Vamplew et al., (2011) proposed 4 testbeds including a deep-sea treasure hunt, MO puddle world, resource gathering and mountain car problem. Natarajan and Tadepalli, (2005) used a modified version of Buridan's ass problem for MOP. Tajmayer (2017) demonstrated a cleaning robot for multi-objective adaptation in RL. The classical multi-objective physical travelling salesman problem (MO-PTSP) is also utilised to explore the search space by (Perez et al., 2015). A wide-ranging standard can be found in "MORL-Glue" that represents a set of multi-objective reinforcement learning framework by (Vamplew et al., 2017b). However, until writing this thesis and to the best of the author's knowledge, there is no such work on dynamic multi-objective optimisation in the reinforcement learning settings. A widespread literature review has been conducted and found that there is a lack of a benchmark that can satisfy the dynamic behaviours of the environment in RL settings. In this research, this limitation has been addressed and proposed a dynamic multi-objective testbed in RL settings. The following

discussions provide essential knowledge to establish the concept and formalise the problem settings for this study.

### **2.3 Intelligent Decision-making Process**

It is essential to delimit the definition of intelligence in the context of decision-making scheme. Therefore, this section provides a brief of the related domain of intelligent applications and comparison of the decision-making schemes.

Intelligent applications or any decision-making scheme and its impact have become an essential part of our daily lifestyles. Recent upgrades in computational power and its processing capabilities encourage developers to develop more intelligent applications (Nick Routley, 2017). Implementation of AI added a new dimension to almost every industry. Most of the applications are using AI to some extent to provide customised user experiences or to make the services even better. These AI-based applications or appliances can assist human to take a better decision or can act according to the set of rules defined by the human. These applications are so smart that they can also think and communicate with some other entities to form a better decision scheme (Gary Orenstein, 2018).

We are using intelligent applications in every domain of our life; be it a complicated one like education, finance, weather, gaming or a simple like one ordering food online. These applications help us in recognising a pattern, classifying different things, clustering, predicting stocks, finding an anomaly, detecting spam and so forth.

The growing industry on smart applications has a clear indication that upcoming days, there will be a lot more applications on medical informatics, business intelligence, emotional intelligence, natural language processing, virtual reality applications and so on (Kasey Panetta, 2017). As a result, there will be a lot of machine learning integrated decision-making scheme in the domain of computational intelligence.

The following applications shown in Figure 2.1 are a glimpse of thousands of intelligent applications which are available in the Apple App Store or Google Play:

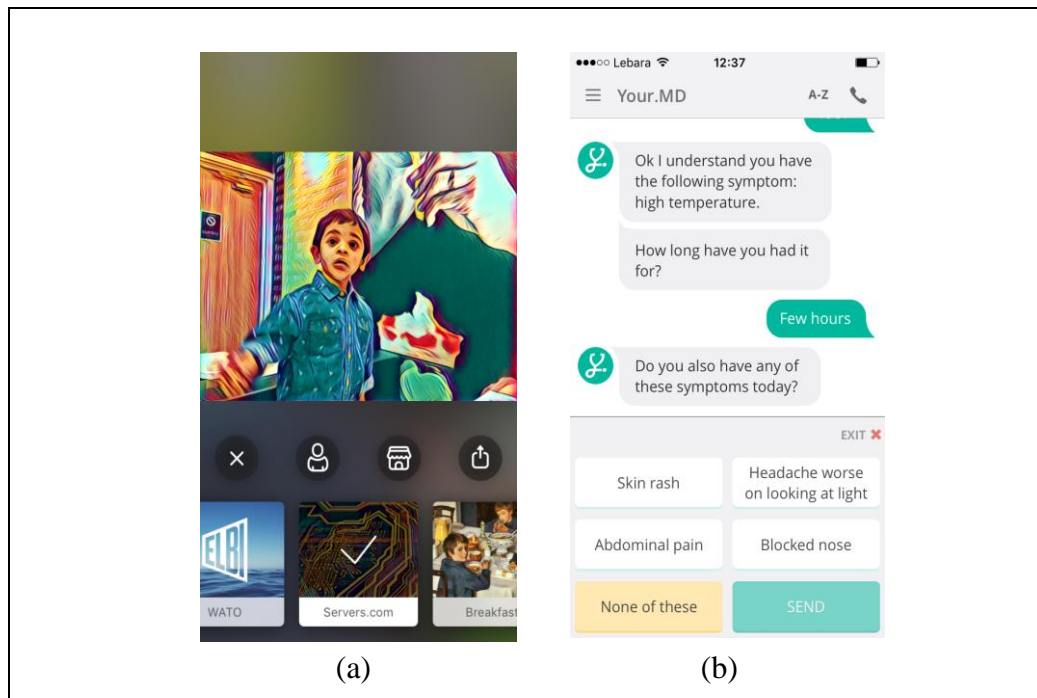


Figure 2. 1: Examples of intelligent applications; (a) is analysing an image and transforming to integrate with famous artists by Prisma, (b) shows a personal medical doctor based on knowledge model and text mining

The following list provides a landscape view in a broader spectrum of the machine learning applications in various branches such as:

- **Medical Informatics:** Classified images and open data have already been used in this domain to produce a good number of AI and ML integrated healthcare applications (Vaishya et al., 2020).
- **Natural Language Processing:** NLP is another very exciting and interesting area of developing applications among the developers. It has a huge potential to increase accuracy and localisation in terms of different languages and dialects (Young et al., 2018).
- **Smart App based financial inclusion and business intelligence:** Traditional business intelligence and the continuous integration of financial inclusion can

only boost this area to enhance socio-economic condition by penetrating in the financial market (Daniel Faggella, 2018).

- Enhancing Cybersecurity: Continuous threats on cybersecurity enhances a common interest in this area among the developers to produce more security and more control in privacy in their developed apps (How Will Artificial Intelligence And Machine Learning Impact Cyber Security?, 2018; Shabut, Lwin and Hossain, 2016).
- Intelligent things: A high-end AI integrated Internet of Things (IoT) devices which connect almost all the appliances and communicate machine to machine usually known as the Internet of Everything (IoE), which have been observed an upcoming booming interest to various intelligent decision-making scheme (Younis, 2018; Bolisetti et al., 2017).
- Digital persona: A digitally represented persona represents or mimic human and assists efficiently from its knowledge and produces an AI-enabled intelligent decision-making system (Carmen del Solar Valdés, 2017).
- Real reality: Blending augmented and virtual reality together with the actual impact in real life using IoT devices could have a phenomenal area of interest in machine learning engineers' especially intelligent decision support system developers (What is Real Reality (RR)? - Definition from Techopedia, 2018).
- Smart City: A rapidly growing area where a smart urban life offers all the necessary and daily life facilities to its inhabitants with an integrated AI-ML based data products and decision-making services (Oktaria, Suhardi and Kurniawan, 2017).

Thus, for the computational researchers, it is obvious that there will be more and more intelligent decision-making tools, technologies or framework integrating machine learning applications. Almost all the sectors that mentioned above require optimisation concerning power, achieving objectives and reducing time-complexity. In this thesis, the scope has been narrowed down focusing on the optimisation area in the domain of intelligent decision-making scheme in DMOPs that needs new knowledge, especially in a data-driven RL setting.



To put it simply, in this study, a decision-making scheme is referred to the context of writing an algorithm (i.e. soft agent) in RL framework that can achieve a compromising solution in a dynamic environment. To get started with the development of intelligent decision-making scheme, a brief review of the decision support system is mentioned in the following sections.

### **2.3.1 Decision support system**

The decision refers to the outcome or a conclusion of a scenario or even to solve a specific question. In other words, a decision-making process involves producing a decision where no alternatives are available or provide suggestions among several alternatives (Hasan et al., 2016; Barraclough et al., 2013). We need to consider or perhaps re-consider a lot of choices in our everyday life to make a decision. Several of them are indirect or direct and several of them depend on different factors or perhaps sub-factors. In almost every case, some steps need to be maintained to reach a final decision to resolve a problem or to answer a specific question.

However, when this process takes place with the help of computer-aided information system, this refers to as a “decision support system- DSS”. The usage of decision support system is increasing in almost every industry such as in business intelligence (BI), human resource management, clinical trial, weather forecasting, financial prediction, disaster management and so forth. Decision support systems (DSSs) are commonly categorised based on their decision making key factors such as data-driven (e.g. IBM Watson Analytics (IBM Watson, 2018), Amazon ML (Machine Learning on AWS, 2018), image analysis based (e.g. optical diagnostics device for cancer detection by MobileODT (MobileODT | The Smart Mobile Colposcope, 2018), communication driven (e.g. Google Docs, Microsoft SharePoint), and knowledge-driven such as Grand Round Table –GRT, a clinical DSS based on knowledge engine (Grand Round Table | Daily Patient Huddle Software | Automate Chart Review, 2018). However, to achieve the aim with defined objectives, this study focused on the data-driven decision support

system. In other words, the proposed algorithm decides based on the data-driven decision-making scheme. A brief description of data-driven decision-making is given below.

Data-driven decision support systems (DDSs) is related to different types of the dataset such as structured (e.g. spreadsheets, relational databases), semi-structured (e.g. XML, email) and un-structured such as audio, video, social media posts like tweets or Facebook status (Brunner and Kim, 2016). Generally, data-driven decision making can be applied where the problem or opportunities are identified and relevant data are collected to be analysed (Janssen, van der Voort and Wahyudi, 2017). The decision-making process includes the following steps as shown in Figure 2.2:

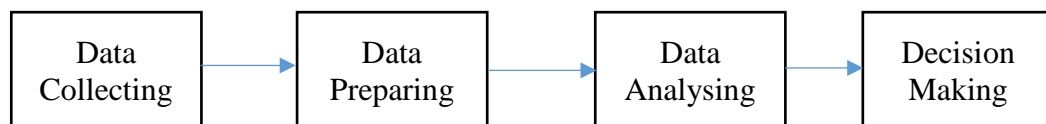


Figure 2. 2: Data-driven decision-making process

However, in this study, the decision support system is based on structured data where data comes from computer-generated environments.

### **2.3.2 Reviewing existing data-driven decision support systems**

In this study, DSSs have been analysed in terms of recent research both in academia and industry. An overview of the popular data-driven decision-making systems is given in the following Table 2.1. Most of these applications have been used in the trial/free version.

Table 2. 1: Data-driven decision support systems

Items	Amazon ML	Microsoft Azure ML Studio	IBM Watson Analytics
<b>Data Type supported</b>	Structured: ✓ Unstructured: ×	Structured: ✓ Unstructured: ×	Structured: ✓ Unstructured: × Data cleaning tool: ✓
<b>Data Cleaning</b>	×	×	✓
<b>Pre-trained model</b>	✓	✓	✓
<b>Collaborative Knowledge Model</b>	×	×	×
<b>Ontology Description</b>	×	×	✓
<b>Decision Interpolation and integration</b>	✓	×	×
<b>Problem Solving Types</b>	Binary Classification: : ✓ Regression: : ✓ Multi-Class Classification: ✓ Pattern Recognition: Text mining:	Classification: ✓ Regression: ✓ Clustering: ✓ Prediction: ✓ Text mining: ✓	Classification: ✓ Regression: ✓ Clustering: ✓ Prediction: ✓ Text mining: ✓
<b>Interactive Visualisation</b>	×	×	✓
<b>Fine-tuner (Controller)</b>	×	×	×
<b>Social Search and social data mining</b>	×	×	✓
<b>Scaling</b>	✓	×	✓
<b>Price</b>	Pay per use	Pay per use	Obsolete

## 2.4 Dynamic Multi-objective Optimisation

To answer the research questions, an overview of the dynamic environment needs to be discussed. The area of dynamic optimisation is concerned with the analysis of various issues that are responsible to make the environment dynamic. Within this section, several issues associated with the time-variant features of the problems will be discussed. Moreover, a review has been carried out regarding the challenges and the future perspective of this domain.

It is essential to distinguish between dynamic and static problems. The uncommon factor of these two environments is the parameters that change over time. In other

words, in the static environment, there are no such changing factors that may influence over time. However, it is opposite to the dynamic environment. A time-dependent problem reacts differently to the changes as time goes by. A detailed definition of the DMOPs has been discussed in Chapter 4.

In the context of the DMOPs, the algorithm's capability depends on whether the algorithm is able to track the changes in a dynamic environment or not. In another sense, the algorithm should be able to trace the behavioural changes over time. This process can be related to the time-linkage property of the problem (Nguyen and Yao, 2009). The second characteristics in the DMOPs, whether the problem can be solved by any meta-heuristic solutions such as evolutionary algorithms. The next phase depends on the problem domain whether it belongs to the discrete or the continuous search space (Lampinen, Lampinen and Zelinka, 1999).

Moreover, predictability and visibility are the other important factors as described in (Nguyen, 2011). The visibility also refers to the frequency of the changes. The other criteria are whether the problem has got single or multi or many objectives that need to be satisfied over time, subject to any given boundaries and the constraints. After that, the target becomes the optimisation goals such as whether the problem is considered as minimisation or maximisation problems.

After that, the necessary constraints are required to set the definition of the problem such as the dependant or independent variables, number of parameters, decision variables, and any related parameters that change over time that has direct or indirect influence in the environment. The next task is to outline the types of problems such as linear or non-linear, episodic or on-episodic, finite or infinite space, stationary or non-stationary problem and so on.

Furthermore, to solve a DMOPs, the characteristics of the problem also needs to be identified such as factors that change over time and the source of changes. This characteristic is also related to tracking the optima. This is usually based on global optima and avoid being stuck in the local optima. Finally, to solve any DMOP, the

evaluation procedure also needs to be defined such as the performance measure for the considered algorithms.

#### **2.4.1 Applications of dynamic multi-objective optimisation problems**

Now, several applications for the DMOPs will be described which can be related to one of the considered test cases in this study such as the water quality problem. Likewise, there are several real-world dynamic multi-objective optimisation problem mentioned in the literature. Helbig and Engelbrecht (2014a) grouped and classified these applications as follows:

Control Problems: The regulation of a lake-river system (Hämäläinen and Mäntysaari, 2001), the optimisation of indoor heating (Hämäläinen and Mäntysaari, 2002), the control of greenhouse system for crops (Ursem et al., 2002).

Scheduling problems: Hydro-thermal power scheduling problem (Deb, Rao N. and Karthik, 2007) and the job-shop scheduling problem (Shen and Yao, 2015).

Mechanical design problems: Design optimisation of wind turbine (Maalawi, 2011).

Apart from these, there are various real-world applications in the domain of dynamic multi-objective optimisation such as in resource allocation (Hutzschenreuter, Bosman and La Poutre, 2009) and routing problems (Meisel et al., 2015) and so on. In addition, Amato and Farina (2005) have proposed an artificial-inspired EA for DMOP in the situation of unpredictable parameter changes. Azzouz, Bechikh and Ben Said (2014) present a multiple reference point-based MOEA (MRP-MOEA) that deals with undetectable changes.

### **2.4.2 Challenges of dynamic multi-objective optimisation**

While reviewing the challenges in the DMOPs, the complexities, as well as conflicts, have been noted carefully. The very first complexity is to identify the definition of the problem itself. Though a lot of definitions have been recommended to categorise the problem, yet, the characterisation of the problem needs to be clearly defined. Therefore, it is needed to be aware of the type of DMOPs.

The very first problem in this domain is the availability of the dynamic benchmark. Nevertheless, to formalise a generic benchmark in the domain of DMOPs are challenging, especially in the combinatorial scenario in which the dynamics rely on various factors that have combined impact. Many researchers have questioned if these benchmark concerns are in fact symbolic of real-world problems and can only cover the simplest form of real-world complexity (Nguyen, 2011). Therefore, these experimental methods are not adaptable or adequate to implement in the real world DMOPs. Thus, most of the solutions are not capable to deal with the challenges of DMOPs (Nguyen, 2011; Nguyen et al., 2013).

The second problem is concerned with the requirement analysis of the real-world problem which may often not aligned with the modelled DMOPs (Apshvalka, Donina and Kirikova, 2009). In addition, sometimes, it is hard to exhibit a problem's requirements and modelling a real-world dynamic problem without proper knowledge and understanding, there would be a high chance that the problem definition is going to be wrong. Therefore, a certain boundary of formulating the problems needs to be systematically evaluated.

In this study, the complexities and challenges have been noted carefully while reviewing the dynamics. Here, the crucial challenges have been observed carefully in the domain of DMOPs. These are given below:

- a. The first one is the lack of standardisation. Though there are many benchmarks in the context of the evolutionary approaches, in the RL or especially in the MORL settings, there is a lack of the standard dynamic multi-objective benchmark. Therefore, it is obvious for the researchers that they measure the performance of the proposed algorithms based on different test functions which show a different set of results in different settings. This case is also true for performance metrics. Consequently, it is not easy to fairly compare the exiting works unless re-implementing and re-evaluating all the algorithms and their performances.
- b. When analysing the dynamic multi-objective optimisation, a time restriction is imposed on the algorithms for checking the convergence as quick as to the optimal PF. However, this restriction may introduce to missing the location of the local or global optima.

As per the above discussion, the benchmark has been created carefully in the context of RL settings to address this problem which has been discussed in detail in Chapter 4.

## **2.5 Reinforcement Learning**

Machine Learning (ML) refers to acquire knowledge or skills through data or experiences (Jordan and Mitchell, 2015). Machine learning has harnessed its applications in various sectors such as decision making and optimisation, medical informatics, fraud and anomaly detection, email filtering, and so on (Das, Dey and Roy, 2015). This learning procedure can be done with or without the supervision of a human. In other words, the way that machine practices to learn can be categorised broadly in three categories as mentioned in Table 2.2.

Table 2. 2: Categories of the machine learning (ML) techniques

Machine learning type	Description	Common algorithms used
Supervised Learning	A prior knowledge is set to get targeted value or outcome by using training data	Decision tree, Naïve Bayes, Ensemble Method, Neural network (supervised), Support vector machine
Unsupervised learning	No prior knowledge of the possible outcome from unlabelled data	Clustering, Gaussian mixture model, Neural network model (unsupervised)
Reinforcement learning	Depends on the state, action and the environment by using positive and negative reward with a discount factor	Markov decision process, Policy gradient, Q-learning, value function approximation, Actor –critic method, Model-based and model-free learning

To begin with, the implementation of the concept of ML, reinforcement learning (RL) is considered because of its applicability in the simulated environment. RL deals with the decision-making problems under uncertainty where the agent learns through the interaction within the environment by taking feedback (Sutton and Barto, 2018; Sutton, 1988). It is a promising way to solve real-world computational problems.

This branch of machine learning is inspired by the trial-error process through experience. RL is selected as one of the top 10 breakthrough technologies by (MIT Technology Review, 2017). It is often described that the concept comes from the attitude of the animal towards learning. Here, the agent interacts with the environment and learns from the environment. This is an established technique now which determines ideal behaviours in a complex environment.

This approach can lead the agent to provide the decision as human-level, but in a complicated environment that human cannot do. For instances, this method was used to a first-ever victory over the human in the game of Go and recent successes in robotics and self-driving cars (Li, 2017). Figure 2.3 shows a typical RL model where an agent takes action by interacting within the environment for each state and earned some rewarding points (Gábor, Kalmár and Szepesvári, 1998).



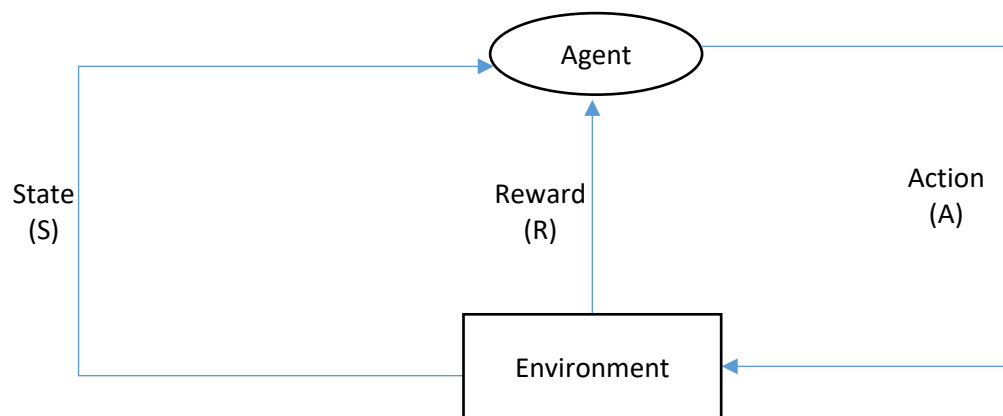


Figure 2. 3: A typical RL model (Sutton and Barto, 2018)

In this section, a brief discussion of the core elements of the reinforcement learning has been provided. The fundamental components of reinforcement learning can be classified as a policy, a reward function, a value function, and a model of the environment. In other words, the RL agent needs to regulate the optimal policy which is the first factor to determine the behaviour. The second important factor is a reward function which defines the goal of RL problem. The aim of the RL agent is to maximise the total reward it receives in the long run. The reward function is also responsible to select which one is the good or bad event for the agent. For instance, if an action is selected by the policy which is determined by low rewards, then the policy could be changed at a later stage when the agent gets the higher reward. The third core component of the RL is a value function whereas a reward function indicates what is good in an immediate sense, and a value function specifies what is good in the long run. The final and fourth component is the model itself. This is something that mimics the actions of the defined environment. For instance, given a state and action, the model is responsible to predict the resultant next state and next reward (Nandy and Biswas, 2018).

To justify the reasoning of using RL approach, it is required to explain when and why to use RL. This branch of ML generally solves a class of problems where the traditional machine learning approaches are not suitable and sometimes not desirable because of the interaction, convergence and the changing environment

or the lack of historical dataset. A common implication of the RL approach is for sequential decision-making where a problem can be solved by a series of actions.

RL is used to solve the problems where multiple solutions are possible. For example, traversing from one route to another. In other words, finding the shortest path such as the travelling salesman problem (TSP) which is one of the classical problems in this domain (Gambardella and Dorigo, 1995). In addition, supervised and unsupervised learning can have decent success considering image and text processing where the uncertainty is limited.

However, there are enormous situations where the certainty cannot be predicted such as in a situation where it is close to impossible to predict the road situation what will be coming next. As a result, it is quite impossible to utilise the supervised and unsupervised approaches in this environment. Therefore, there is no way to predict the road that what happened next and thus, the algorithm needs to be flexible, adaptable to comply with the different, dynamic and unexpected conditions. Another example can be a prodigious complex environment such as control of an arm of a robot or moving unmanned cars. Though this can be hard-coded with the traditional if-elseif-else conditions, the process may take abnormal time to be trained and there is a high chance of establishing vulnerable AI agent (Luke Dormehl, 2018). Contrarily, a self-learning agent can be a time and effort saver. In addition, the RL technique is also helpful to know what others are doing such as enemy or opponents' strategy to learn the next steps. In a nutshell, where the policy or creativity needs to be followed, the RL can be a righteous choice for the AI engineers (Ravichandiran, 2018).

While praising the RL approach, it has also the downside that needs to know for further exploration. There are several places where the RL approach will not be suitable compared to traditional machine learning approaches. Therefore, RL might not be an ideal choice of solving the problems which can be resolved easily by supervised and unsupervised learning. In such cases, the RL agent will not be as effective as traditional machine learning techniques and applying RL may have

little sense in these contexts. Furthermore, RL is usually computationally expensive and challenging to use (Collins and Frank, 2012). Besides, the RL agent may lead the decision makers to the wrong point if the agent does not explore all the states in the environment. Generally, the RL agent requires piles of computational power to be trained and to build the environment. All in all, it truly depends on the stakeholder's choice and developers' freedom to use the technique which can bring the best solution according to the nature of the problems.

### 2.5.1 Multi-objective reinforcement learning (MORL)

Multi-objective reinforcement learning (MORL) was proposed by (Gábor, Kalmár and Szepesvári, 1998). MORL deals with the decision-making problems in uncertain situations where the agent learns by interacting and taking feedback within the environment (Sutton and Barto, 2018). It is a promising way to solve the problem which has more than one objective. Figure 2.4 shows the basic structure of a multi-objective reinforcement learning (MORL) setting where the agent interacts within the environment and tries to find out the optimum solutions based on the pre-defined objectives.

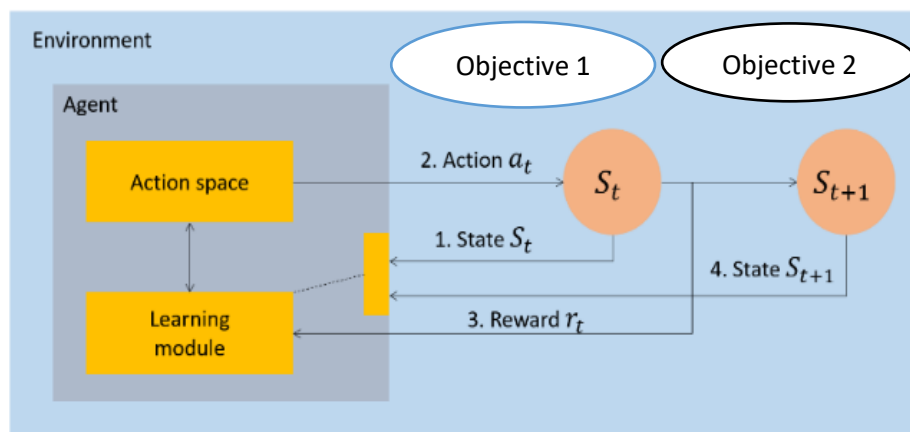


Figure 2. 4: A typical setup for multi-objective reinforcement learning environment

The major difference between single objective and multi-objective reinforcement learning depends on the use of reward mechanism; such as scalar reward and vector reward which are used for single and multi-objective problem respectively (Liu, Xu and Hu, 2015). In the MORL, the agent needs to learn either all Pareto optimal policies or identify a single policy that best matches a pre-specified trade-off between the objectives.

Vamplew et al., (2017a) proposed that Multi-objective reinforcement algorithms which can be classified into two broad categories. The first one is the multi-policy approach where the agent learns multiple policies to approximate the Pareto front or a subset of the Pareto front. While, in the second one, the agent learns a single policy which best satisfies the compromising solution between objectives. In addition, an MORL agent differentiates multiple policies in three broad types namely deterministic stationary, stochastic policy and non-stationary policy (Vamplew et al., 2017a).

Furthermore, a constrained MORL method was used to optimise the average transmission delay in a cognitive radio network proposed by (Zheng et al., 2012). In the area of decision support system, MORL is used in medical informatics as mentioned in (Lizotte, Bowling and Murphy, 2010). Moreover, a solution to find an optimal path has been investigated in the multi-objective stochastic environment by using MORL as described in (Tozer, Mazzuchi and Sarkani, 2017). In this study, MORL algorithm is formulated to solve multi-objective Markov decision process (MOMDP), which was pioneered by (Roijers et al., 2013).

Consider the following example, where there are two different objectives  $\min(f_1)$  and  $\max(f_2)$  represent two different conflicting objectives (Herrmann, 2015). The red area in Figure 2.5 (a) shows the true PF based on the compromising solution. In other words, the red-area illustrates the non-dominated solutions considering the objectives 1 and 2. If the Pareto area is considered in a shape-dependent way, it will look like Figure 2.5 (b).

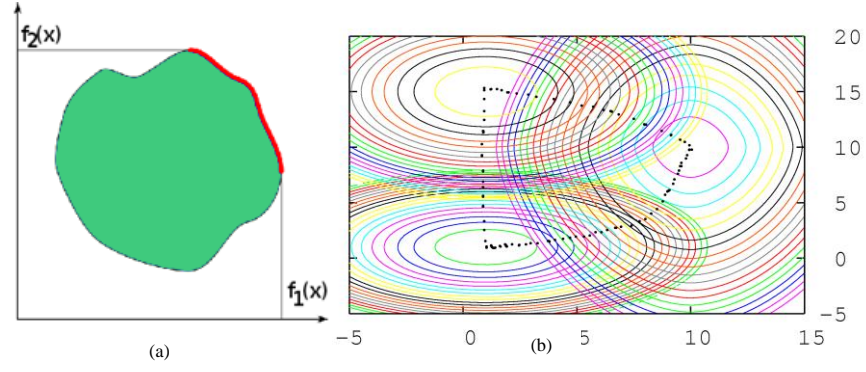


Figure 2. 5: Multi-objective optimisation in MORL (a) True PF (b) PF covered by maxima in a shape-dependent way (Herrmann, 2015)

### 2.5.2 Reviewing existing approaches of MORL

There are two strategies to solve the MORL. One is scalarised approach, and the other one is the Pareto approach. In the scalarised approach, the MORL agent looks for a single policy that optimises the combination of the rewards. This approach looks for the preferable reward or weighted sum of the rewards that need to be selected in a particular state. On the other hand, the Pareto approach finds multiple policies that cover the Pareto front.

Multiple reward signals are parametrised corresponding to the Pareto front. In case of a non-connected Pareto front, non-Pareto optimal solutions may be found. A parametrised combination of multiple reward signals is used with different parameters in different runs to address dissimilar points.

The set of all non-dominated solutions obtained in this way provides the Pareto front. The agent may change the parametrisation according to progress on each of the goals. Table 2.3 shows the different MORL approaches (Vamplew et al., 2011).

Table 2. 3: Different MORL approaches

MORL approaches	Description
Single-policy approaches	Weighted sum approach
	W-learning approach where each objective has their own recommendation and finally the largest value-based objective gets selected
	Analytic hierarchy process (AHP) based approach that used for multi-criteria selection
	In the ranking approach, partial policies are used to select the action
	In the geometric approach, geometric conditions are used for a synthetic objective function
Multiple-policy approaches	The convex approach which learns multiple policies in the objective space
	The varying parameter approach performs any single-policy that runs multiple time but with different parameters and objective values

Furthermore, converting a multi-objective problem into a single objective one will be the simplest way to use RL algorithms. Inside MORL, the profits are provided in vector rewards for every objective. On the other hand, a single goal RL has scalar incentives. Moreover, a multi-objective task could be minimised to a single objective via scalarisation.

Nevertheless, the algorithm can be changed or even modified to recognize the values for every objective to produce a better outcome. MORL algorithms are accomplished long-range average incentives while staying in the externally defined ‘target’ region. As a result, non-stationary policies have created the role of the present average rewards connected to the target region. Pareto dominance is usually called a ‘good’ compromise in this regard.

As mentioned earlier, this rectifies that an objective is said to be non-dominated if it is better at least in objective and not worst in any other objectives. Hence, the aim is producing an estimated set of a true Pareto front. It is already been pointed out that the majority of the current MORL solution is able to make a single

solution. Although several authors have conducted research on producing different solutions using a single objective in the case of the multi-objective problem. One of the solutions proposed by a policy gradient MORL algorithm which has created from the independent use of RL, is non-negative with respect to other goals (Parisi et al., 2014). To study the dynamic multi-objective optimisation, there are two major approaches in the literature such as synthetic or real-world problem (Azzouz, Bechikh and Ben Said, 2017).

In this study, a simulated environment and a real-world scenario have been taken into consideration. As cited earlier, a rising approach of solving MO problems is using RL technique. This area gets major concentration after the successful implementation by DeepMind to solve the Atari Games 2600 as a superhuman level with only raw pixels and scores as input (Mnih et al., 2015). To achieve human-level expertise on 49 classic Atari games, they used a single architecture and showed how the agent can successfully learn control policies in a range of different environments with minimal prior knowledge. It uses the same algorithm, network architecture and hyperparameters on each game. However, any relevant literature was not found that can satisfy the dynamics of type I, II, III and IV as mentioned in (Farina, Deb and Amato, 2004) in a MORL setting. According to the above discussions, it is inevitable that to solve the problems in the dynamic multi-objective environment, a vector reward-oriented approach is needed in the MORL settings which represents a multi-policy approach.

There are several ways to solve multi-policy RL problems (Sutton and Barto, 2018). They are as follows:

- a) Markov Decision Process
- b) Dynamic Programming
- c) Temporal difference learning
- d) Q Learning
- e) Deep Learning

f) Monte-Carlo Tree Search (MCTS)

Among these algorithms, in this study, the Markov decision process (MDP) is used to analyse the states and necessary actions for each state. The reason behind using MDP in this study is to clarify each state and formalise the environments in the simulated environment to find out the optimal policy with value approximation function.

## **2.6 Markov Decision Process (MDP)**

Markov decision process (MDP) is the widely studied model to analyse dynamic and stochastic system (Bertsekas and Tsitsiklis, 1996) and used to solve MORL problems. This framework provides a standard model to design the system with probabilistic, nondeterministic and controlled behaviour which helps to analyse the states and necessary actions for each state. In this study, an MDP helps to structure the dynamic environment to identify the non-dominated policies with Q point which is often denoted by Bellman's equation (Understanding RL: The Bellman Equations, 2017).

Furthermore, MDP is used in a wide range of real-world tasks such as robot control (Kober, Bagnell and Peters, 2013), game playing (Szita, 2012), clinical management of patients (Peek, 1999), military planning (Aberdeen, Thiébaux and Zhang, 2004), control of elevators (Crites, Crites and Barto, 1996), power systems (Glavic, Fonteneau and Ernst, 2017), and water supplies (B. Bhattacharya, A.H. Lobbrecht and D.P. Solomatine, 2002) and so forth. MDPs are solved by planning a model inside the MDP (e.g., dynamic programming methods), (Bellman, 1958) or by learning through interaction with an unknown MDP (e.g., via temporal-difference methods), (Sutton & Barto, 2018). MDP follows two approaches such as single policy and multiple policies by using Pareto optimal values.



### 2.6.1 Single objective Markov decision process

In a single objective RL setting, an MDP is mostly defined with a tuple of state, action, transition, reward, probability distribution and discount factor. It is often defined as follows (Feinberg and Shwartz, 2014):

$$\langle S, A, T, R, \mu, \gamma \rangle$$

Where  $S$  and  $A$  represents a finite set of states and a set of actions respectively.  $T$  represents transition function,  $R$  defines the reward function,  $\mu$  is the probability distribution over initial states,  $\gamma$  represents a discount factor to signify the importance of short and long-term rewards.

In this single objective setting, the objective of the agent is to maximise the expected reward function  $R_t$  by a suitable policy  $\pi$ . Besides, for the policies which determine an agent's action, there will be a single policy or multiple policies with the same return. In the single objective MDP, an optimal policy  $\pi^*$  is responsible to find out the maximum expected values for all the states. It is to be noted that, a policy is Pareto optimal or non-dominated if the policy is not dominated by any other policy.

### 2.6.2 Multi-objective Markov decision process (MOMDP)

An MOMDP is an extension of MDP where the reward function receives a vector of  $n$  rewards  $\vec{r}$  that represents each objective (Mossalam et al., 2016). The solution of the MOMDP is a set of policies that contains at least one optimal policy for each possible preference. In an MOMDP, a policy can outperform with one objective. However, the performance can be the worst or inferior compared to other objectives. A policy which depends on the current state is called the stationary; otherwise, it is non-stationary. On the other hand, a policy that chooses the same action in the same given condition is called deterministic if not it is stochastic. This can be summarised with Pareto dominance relationship such as a policy dominates on the other objectives if it is superior at least in one objective or equal or not

worse than any other or all objectives (Lwin, Qu and Kendall, 2014). Preferably, a non-dominated policy is more acceptable than a dominated policy (Algoul et al., 2011).

In an MOMDP, a state value function  $V^\pi$  specifies the expected discounted return when following a policy  $\pi$  from the initial state  $S_0$ . This can be formulated as in Equation 2.1 (Ruiz-Montiel et al., 2017):

$$V^\pi = E_\pi \{\vec{R}_t\} = E_\pi \sum_{k=1}^n \{\gamma^k \vec{r}_{t+k+1}\} \dots\dots\dots (2.1)$$

Here, a vector reward  $\vec{r}$  received after  $k$  steps is worth  $\gamma^{k-1}$ ,  $E_\pi$  represents expected policy and expected discounted return  $\vec{R}_t$  which needs to be maximised. The policy basically relies on the definition of the Pareto optimality or dominance between vectors. For example, a policy  $\pi_1$  is dominated over  $\pi_2$ , if the value of  $V^{\pi_1} > V^{\pi_2}$ . Similarly,  $\pi_1$  is dominated or equal to  $\pi_2$ , if the value of  $V^{\pi_1} \geq V^{\pi_2}$ . There may be many other non-dominated policies which are optimal depends on the desired trade-off between policies.

For stationary policies, the state value function is defined as in Equation 2.2 (Ruiz-Montiel, Mandow and Pérez-de-la-Cruz, 2017),

$$V^\pi = E_\pi \{\vec{R}_t\} = E_\pi \sum_{k=1}^n \{\gamma^k \vec{r}_{t+k+1} | S_t = s\} \dots\dots\dots (2.2)$$

However, in this study, MOMDP is used with deep learning structure to solve DMOPs. In this case, an obvious question might be — why do we need deep reinforcement learning to solve the dynamic multi-objective optimisation problem? To answer this question, let us have a close look at the deep reinforcement learning architecture in the section below where the agent learns itself like a human to achieve successful strategies based on a deep neural network. This leads to the highest long-term rewards that are constructed and learnt by interacting with the environment.

## 2.7 Deep Reinforcement Learning (DRL)

The rise of DRL is very recent. Deep learning or deep neural network has been predominant in the reinforcement learning area in the last several years. In the deep layer, artificial neural network (ANN) is tailored to mimic natural neural networks using a computing procedure (Haykin and Simon, 1994). Generally, an ANN has the configurations as shown in Figure 2.6 of the topology of X-H-O-T node where X and O are inputs and outputs, respectively. Besides, H and T represent hidden nodes and the target output, respectively. All these layers contain a number of interconnected neurons (i.e. processing units).

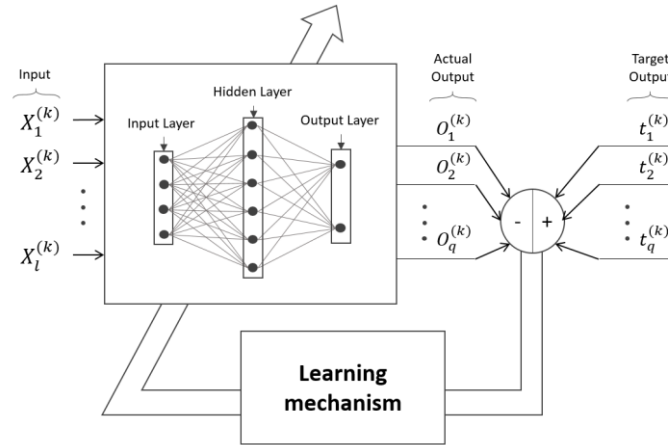


Figure 2. 6: Schematic diagram of the learning mechanism based on ANN (Fakhreddine and Clarence, 2004)

Deep RL has significantly reduced the reliance of the domain knowledge and enables highly efficient feature engineering which is usually time-consuming, over-specified or incomplete (Li, 2017). Many positive outcomes in DRL are based on previous studies of RL in high dimensional problems that is due to the learning to low dimensional feature representation and the effective feature approximation components of neural networks.

Besides, DRL can solve the highly complex computational problems. Since the empirical return of trajectory is essential for the computation, the ensuing gradients involve a higher adjustment. To reduce the level of variance, impartial estimates

that include less distraction needs to be introduced. A common way of doing this is to deduct a baseline which means weighing revisions by an advantage rather than the original return (Li, 2017).

Figure 2.7 shows the timeline for the evolution of the DRL (Laura Schneider, 2018). It is clearly noticeable that the process of DRL has not started recently (Gil Press, 2018). The whole process was revolutionised when AI came to flourish in the decade of 1940 to 1975 (Piero Scaruffi, 1996). After 2000, there was a revolution of computation power to compute the complex operations with the help of deep learning. Today's deep reinforcement learning has achieved great success around 2015 by DeepMind using the convolutional and recurrent neural networks (Schmidhuber, 2015; Mnih et al., 2015).

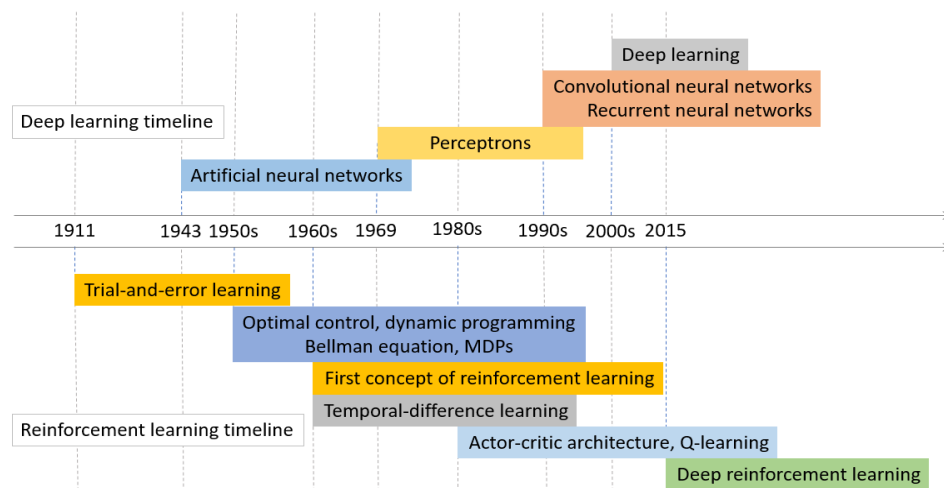


Figure 2. 7: Timeline for the evolution of deep reinforcement learning

We witnessed breakthroughs, like deep Q network (Mnih et al., 2015), AlphaGo (Silver et al., 2016), unsupervised reinforcement and auxiliary learning (Jaderberg et al., 2016; Mirowski et al., 2016). The successful implementation of deep reinforcement learning has appeared in different areas such as:

- Power System: DRL based AI decreases Google data centre bill by 40% (DeepMind, 2016).
- Robotics: Black-Box Data-efficient Policy Search for Robotics (Chatzilygeroudis et al., 2017).
- Games: Human-level Control through Deep Reinforcement Learning (Mnih et al., 2015).
- Spoken dialogue system (Su et al., 2016).
- Information extraction (Narasimhan, Yala and Barzilay, 2016)

### **2.7.1 Basic architecture of deep reinforcement learning**

This section highlights the fundamental architecture of DRL. Figure 2.8 shows a multi-objective deep reinforcement learning model where an agent takes an optimal action (i.e. policy) for a state in an environment and earns reward points (e.g. vector rewards for multi-objective cases). This model is formulated by the Q network, target network, emulator and experience replay.

Usually, the Q network is comprised of the convolutional layers, fully-connected layers and the output layers. This architecture is different based on the different setups such as in the GoogleNet, AlexNet where the number of layers and their arrangements are different (Siddharth Das, 2017). This is varied based on the internal neural network structure and the weight-bias setup for a certain deep network.

In the experience replay, the samples of transitions are stored randomly where the agent emphasises the important experiences based on the actions. The emulator is responsible to generate the episodes in memory. The learning module is updated after each episode and the target network updates based on the updated Q value.

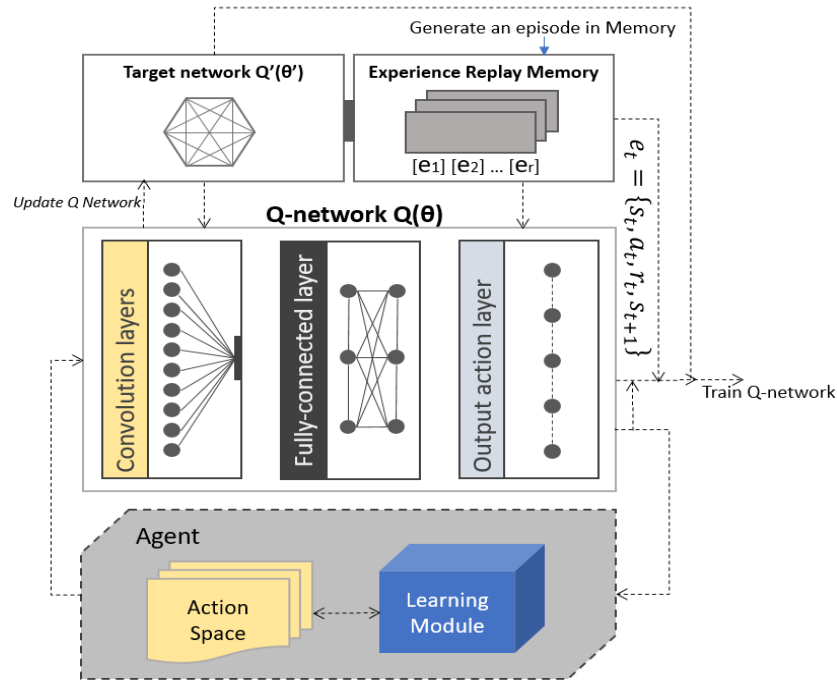


Figure 2. 8: The system architecture of deep reinforcement learning

From the above Figure 2.8, it is also observable that the target network  $Q'(\theta')$  is iterating until it gets the highest expected reward over time that can easily fit within the dynamic environment according to our requirements.

The agent learns the environment based on the traversing and the interaction within the environment. The trajectories for heuristic investigation may conduct an exhaustive search in the search space. According to (Sutton and Barto, 2018), the agent plans the strategy to develop the optimised policy. Inside the DRL structure, the agent focuses to access and identify the underlying variety of the environment. In the dynamic environment, this is the obvious case since the environment gets changed based on different fluctuating factors over time.

Nevertheless, interactions within the environment help the agent to understand the truly worth features, policies, and furthermore a model. As we have considered a Model-free RL strategy, therefore, the agent tries to find out the optimal policy directly from the interactions within the world. However, learning a model by the off-policy agent may introduce unwanted complexities. For instance, the planning

of the agent depends on the output of the convolutional neural networks (CNNs), allowing them to find out immediate actions from raw input or substantial dimensional of visual inputs. Therefore, DRL is based on instruction heavy neural networks with several convolutional layers to acquire the best policy. DRL is used in different real-world applications such as in healthcare (Liu et al., 2017), smart transportation system (Schultz and Sokolov, 2018), smart grid (Panov, Yakovlev and Suvorov, 2018), etc. Besides, DRL has got some interesting applications such as music generation (Briot, 2018) and procedural content generation (Justesen et al., 2018).

In a nutshell, DRL has the following advantages:

- It is completely autonomous,
- Does not need prior knowledge of the environment,
- Performs online and offline,
- Learns as well as optimises autonomously,
- DRL can provide various optimisation techniques and
- Various reward capabilities help the agent to distinct policies, without the demand of re-engineering the process.

On the other hand, the disadvantages of DRL can be listed as follows:

- Black-box optimization,
- Time-consuming and requires high-end devices to perform and
- Being model-free has the cost of being highly computationally expensive.

### **2.7.2 Challenges in deep reinforcement learning**

While working with the DRL in this thesis, several challenges have been observed. It is instructive to emphasize some challenges faced in DRL for optimisation which may be helpful to the readers for future references and further investigations. They are given below (Li, 2017):

- Choosing a learning rate,
- Defining an annealing schedule,
- Avoiding suboptimal minima,
- Setting up the vector reward for the multi-objective environment,
- The observations of the agent can have overestimation or underestimation and
- Can suffer from temporal credit assignment problem (Sutton, 1984).

After reviewing DRL literature, a list of the new scopes of DRL research is given below:

- Deep layer fine-tuning to get more insights and meaningful behaviours of the deep Q networks,
- Self-learning and human-agent teamwork for the optimisation problem and
- Off-policy and model-free integration for an on-demand service by the agent.

## **2.8 A General Framework for Deep Q Network (DQN)**

In this section, one of the core components in this research will be discussed which is the general framework of the DQN. At first, Q learning and the basics of DQN network will be discussed. After that, the different modification procedure of the Q learning mechanism will be highlighted. Finally, the prioritised experience replay and the comparisons of the different DQNs will be explained to get a deeper understanding of the proposed PQDQN algorithm in Chapter 5.

### **2.8.1 Q learning**

Q learning (Watkins and Dayan, 1989) is a model-free technique in a RL setting which is used to learn an optimal  $Q(s, a)$  for the agent in an MDP  $(s, a, r, s', \pi)$ . It is also off-policy, optimised value function and most importantly, this fits for the simulated environment. In addition, the agent based on the Q learning learns a



unique policy in the case of the single objective scenario such as when rewards are scalar. This scalar-values  $Q(s, a): S \times A \rightarrow \mathbb{R}$ , represent the expected accumulated reward when following a given policy after taking an action  $a$  in state  $S$ . The action  $a$  is selected by the policy in each state. This selection is usually expressed by  $\operatorname{argmax}_a Q(s, a)$ . In a single-objective MDP, a deterministic stationary optimal policy is very likely to be found. In this case, the expected return is usually an undiscounted finite sum of the scalar rewards until the agent reached the terminal states. Regarding the computational complexity of the Q learning, the worst-case complexity becomes  $O(n^3)$  if no state space has no duplicate actions as reported by (Koenig and Simmons, 1993).

### 2.8.2 Basics of a deep Q network (DQN)

In Q learning, the RL agent interacts within the environment. This decision-making procedure has been done through a sequence of steps. Thus, it has formed a sequential decision-making process. This decision-making process starts at the first step which is selected by the agent arbitrarily and the traverse ends when the agent reaches the terminal state. The agent observes its current state  $S_n$  while the agent is in  $n^{th}$  state. Then the RL agent performs an action  $a_n$  and observes its current state  $S'$  receives an instant reward  $r_n$  and adjusts the value  $Q(s_n, a_n)$  using a learning factor  $\alpha_n (0 < \alpha_n \leq 1)$ . The updating expression of the scalar Q-learning algorithm can be expressed using the Bellman equation (Bellman, 1958). However, in the context of MORL, the Q learning needs to be tweaked so that it can work with the vector rewards. Thus, the equation requires to be extended to handle vector operations. Therefore, in the finite horizon or an episodic environment, the reward function  $\vec{R} = S \times A \times S'$  is a vector of  $n$  rewards rather than a scalar with an element for each objective. Similarly, the reward function is also the vector value such as  $\vec{r} = \vec{R}(s, a, s')$  as defined earlier by Equations 2.1 and 2.2 (Ruiz-Montiel, Mandow and Pérez-de-la-Cruz, 2017).

Moreover, in the vector reward space, an action  $a$  in state  $s$  under any non-dominated policy can be denoted as  $\vec{Q}^*(s, a)$ . Here, the fact is the obtained reward is a vector where the agent does not learn a single policy but a set of policies at the same time. The values learned in this scenario can be denoted as  $\vec{Q}(s, a)$  of vectors, which are used to estimate the optimal  $\vec{Q}^*(s, a)$  sets.

This can be denoted by Equation 2.3 which is inspired by the enhanced proof of the Bellman equation mentioned in (Gross, 2016):

$$\vec{Q}_n^*\{(s, a), t\} = \begin{cases} (1 - \alpha_n)\vec{Q}_{n-1}(s, a) + \alpha_n[\vec{r}_n + \gamma V_{n-1}(s'), t]; & \text{if } s = s_n \wedge a = a_n \\ \vec{Q}_{n-1}\{(s, a), t\}; & \text{otherwise} \end{cases} \dots (2.3)$$

$$\text{Where, } V_{n-1}(s) = \max_{a \in A} \vec{Q}_{n-1}\{(s, a), t\}$$

### 2.8.3 Q-function modifications in DQN

In most of the cases, the DQN works based on the modifications of the Q value(s) (Li, 2017) . This depends on the different strategies of Q-value exploitations. In this study, the major part of the proposed algorithm deals with the Q value approximations and updating the Q table. This is a standard procedure for updating the Q value in DRL based solutions as discussed earlier. Hence, a brief discussion of the Q-value variations will be deliberated which is necessary to follow the proposed framework.

While reviewing the literature, it is observed that Q value is the core characteristics to approximate the value of the target network. It has been proved by (van Hasselt, Guez and Silver, 2015) that Q learning overestimates the expected return in the single estimator setting. The possible solution for this over-estimation is double-Q learning by offering a better estimate using a two-fold estimator.

Nevertheless, decomposing the Q function into meaningful features is another common way to solve the overestimation problem. It works as like as the divide and conquers method. In this study, the decomposition procedure has been followed in the proposed algorithm. In this scenario, the RL agent gets the Q value from different DQNs and find out the best one from the Q table to update the network. However, in the context of the dynamic environment, at each time step  $t$ , the agent receives a reward  $r_t$  in a state space  $S$  and selects an action  $a_t$  from an action space  $A$ . Thus, the agent finds the optimal policy based on these selections from  $(a_t|s_t)$ . Consequently, the agent forms its behaviour for a particular episode. In other words, the agent's target is to maximise the long-term returns from each state. Figure 2.9 illustrates a decomposition structure for formalising a Q value in a DQN architecture (Lin et al., 2017).

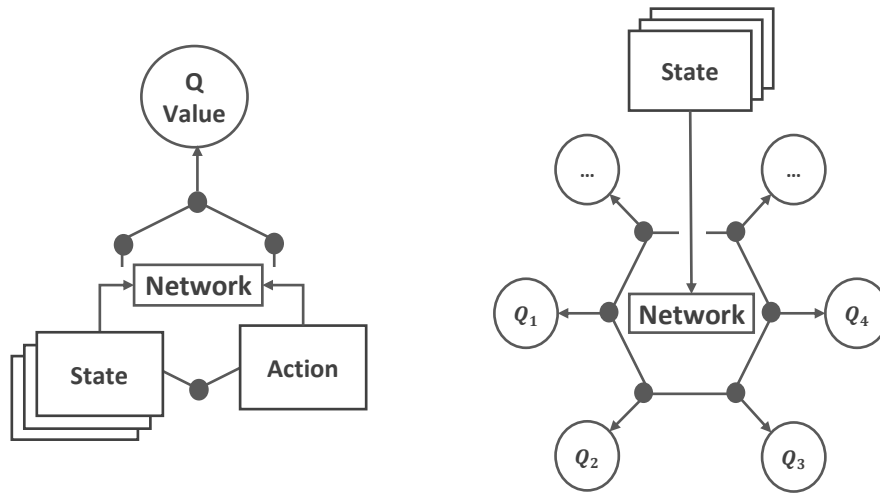


Figure 2. 9: Q value selection in a Deep Q network

In this study, to utilise the decompositions architecture of the DQN, prioritised experience replay or the replay memory (RM) has been used. Therefore, the following section explains prioritised experience replay memory with a brief discussion on the duelling architecture.

#### 2.8.4 Prioritised experience replay for DQN

In the DQN, the sampled experiences which are observed by the agent needs to be stored. Therefore, the experience transitions are uniformly sampled and stored in the replay memory. This process is done regardless of the significance of experiences (Schaul et al., 2015). The purpose of using the experience replay memory is to iterate the important experiences and prioritise them.

These experiences ensure the transitions that are important in an episodic environment. It is worth mentioning that the priority may be biased or the sampled might be insufficient if the agent is not capable to traverse all the nodes. Therefore, the  $\epsilon$ -greedy mechanism has been implemented to ensure the agent has traversed all the nodes to sample the transitions more appropriately.

Moreover, the agent calculated the transitions using temporal difference (TD) errors that helps the agent to avoid bias (Ruben, 2016). The following Figure 2.10 illustrates the schematic diagram for the experience replay.

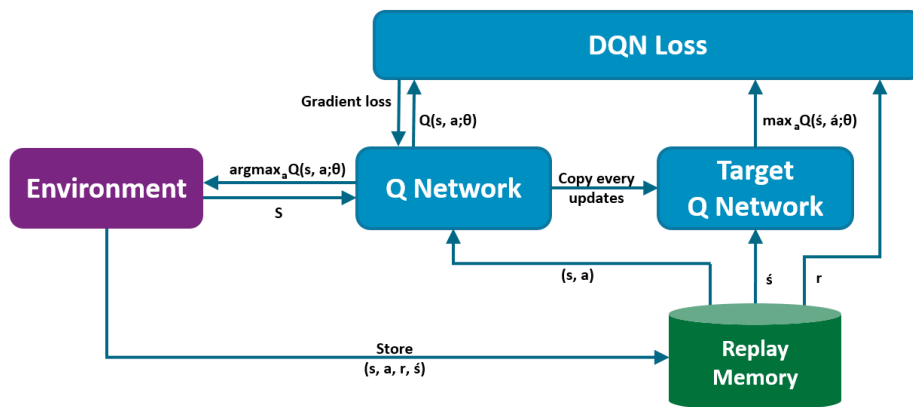


Figure 2. 10: Connection of a prioritised experience replay memory in a DQN

### 2.8.5 Reviewing different deep Q networks (DQNs)

A comprehensive review has been conducted in this thesis to choose the appropriate DQN structure. As a result, a comparison has been carried out to identify the suitability of the different DQN techniques in terms of strengths and weaknesses of the considered algorithms. This section describes the common DQN algorithms' aim, description, advantages and disadvantages. Table 2.4 reviews the policy optimisation and actor-critic methods.

Table 2. 4: Reviewing policy optimisation and actor-critic approach

<b>Policy Optimisation</b>	
Aim	To optimise policies in an episodic or stochastic environment where values are yet to be explored. It is to be noted that a policy $\pi$ optimisation can be simpler than finding Q and value V.
Description	Step 1: Set the policy parameters Step 2: Fix policy rules in accordance with policy variations Step 3: Compute gradient to determine policies Step 4: Find the optimum policy
Pros	a) Perform better in converging properties b) Can learn stochastic process
Cons	a) Usually, converge to a local rather than in global optimum b) Challenging to find out a good estimator policy gradient c) Validating policies are typically inefficient and high variance
<b>Actor-critic Method</b>	
Aim	To seek a parameter vector which maximizes the return by using a policy gradient approach where an actor deals with the policy and critic evaluate the current policy.
Description	Step 1: Initialise state and policy factors Step 2: Set critic learning rate and actor learning rate Step 3: Get sample reward, transitions and actions Step 4: Observe reward and evaluate policies prescribed by the actor Step 5: Repeat until the terminated state
Pros	a) Reducing high variance and effective in high dimension b) Evaluated policy by critic
Cons	This method may introduce bias

Now, the reviews of the different DQNs have been described to get a depth understanding. Table 2.5 highlights the different DQN's aim, merits and demerits.

Table 2. 5: Reviewing different DQNs

<b>DQN</b>	
Aim	Define function $Q(s, a)$ which refers to the maximum discounted future reward for each action $a$ in state $S$ and optimise continuously from that point.
Description	<p>Step 1: At first, define <math>Q</math> value and policy <math>\pi</math> which set the rules of how to choose different actions in a particular state.</p> <p>Step 2: After setting the <math>Q</math> value, pick the highest <math>q</math> value and fix <math>Q</math> function using the Bellman equation (Bellman, 1958).</p> <p>Step 3: Sample random transitions from a set of temporary experiences</p> <p>Step 4: As <math>Q</math>-function converges, the network gets more appropriate <math>Q</math> values and thus, exploration decreases</p> <p>Step 5: Train the <math>Q</math> network until a terminal state</p>
Pros	<p>a) It performs faster while training the network considering it uses random minibatches from temporary memory instead of recent transitions.</p> <p>b) In the episodic environment, if the space is too big then <math>Q</math>-table can be replaced instead of the deep neural network.</p>
Cons	As the agent follows greedy approaches to fix the $Q$ value, it is often may lead to less optimised policy and increase time complexity.
<b>Double DQN Algorithm</b>	
Aim	The aim of double $Q$ -learning is to lessen overestimations which are often caused by the standard $Q$ learning algorithm by decomposing the <i>max</i> operation in the target into action selection and action evaluation
Description	<p>Step 1: Take inputs of empty and replay buffer, initialise the network parameters, training batch size and the target network</p> <p>Step 2: For each episode, initialise frame sequences</p> <p>Step 3: Set state and sample actions</p> <p>Step 4: Append new frame and delete the old one</p> <p>Step 5: Sample minibatches and set the target network</p> <p>Step 6: Compute the gradient descent step with loss and replace target parameters for each step.</p>
Pros	<p>a) It performs better to reduce observed overestimations compared to <math>Q</math> learning.</p> <p>b) Double DQN finds better policies and it is more stable and reliable in terms of learning.</p>
Cons	Single stream double DQN performs worse than DuDQn.

<b>Dueling Q Network (DuDQN)</b>	
Aim	It is a model-free RL algorithm aims to provide two separate estimators such as state-value function and state-dependent action function to generalise learning across actions
Description	<p>Step 1: Set a stream of a scalar <math>V(s; \theta, \beta)</math> that represents a fully-connected layer. The other stream of output vector <math>A(s, a; \theta, \alpha)</math>. <math>\theta</math> represents the parameters of the convolutional layers. The <math>\alpha</math> and <math>\beta</math> are the parameters represented based on these two streams of fully-connected layers.</p> <p>Step 2: Then it can be integrated with Double DQN [shown above] or prioritised experience replay</p>
Pros	<p>a) It provides better results for policy evaluation.</p> <p>b) The architecture is able to learn the states which are valuable or not. This is achieved without learning the effect of each action for each state.</p> <p>c) It uses two separate streams so that Q values can be combined that leads to producing a single Q function. Consequently, this can be used to train some other RL algorithms such as DDQN and SARSA.</p>
Cons	It performs better than standard Q network only with the large set of actions.
<b>Continuous DQN</b>	
Aim	It aims to provide better performance with normalised advantage functions as an alternative of policy gradient and actor-critic method for continuous actions.
Description	<p>Step 1: Initialise a normalised Q network, a target network and a replay buffer</p> <p>Step 2: For each episode, initialise the random process for action exploration</p> <p>Step 3: After getting an initial observation, select action and store the transitions</p> <p>Step 4: Sample random minibatches from replay memory</p> <p>Step 5: Update the target network by minimising loss until the terminal state</p>
Pros	It substantially improves performance on a set of simulated robotic control tasks.
Cons	Continuous DQN performs worse than DDPG for finding better policies in the continuous domain spaces.
<b>Deep Deterministic Policy Gradient (DDPG)</b>	
Aim	It is a model-free RL algorithm that can learn competitive policies using low-dimensional observations and can often achieve good policies direct from pixels with the same network structure.
Description	<p>Step 1: Initialise critic network and an actor with weights</p> <p>Step 2: Initialise a target network and a replay buffer</p> <p>Step 3: For each episode, initialise a random process for action exploration</p>

	<p>Step 4: For each time, select actions according to the current policy and exploration noise</p> <p>Step 5: Store transitions and sample a random minibatch</p> <p>Step 6: Update critic, actor and target network</p>
Pros	<p>a) It can learn better policy among several competitive policies</p> <p>b) DDPG can treat the problem of exploration in the continuous spaces independently from the learning algorithm.</p>
Cons	Usually, it requires a huge number of training episodes to get optimum solutions.
<b>Asynchronous N-step Q Learning (ANSQ)</b>	
Aim	To design a RL algorithm that can train the deep neural network policies faster and reliably with minimum resource requirements
Description	<p>Step 1: Initialise a counter, a target network with thread-specific parameters and network gradients</p> <p>Step 2: Select actions using exploration policy until the state reaches its final state or up to <math>t_{max}</math>.</p> <p>Step 3: Compute gradients for each state-action pair for n-steps of the Q-learning updates</p> <p>Step 4: Update asynchronously global shared parameter vector <math>\theta</math></p> <p>Step 5: The accumulated updates are applied in a single gradient step</p>
Pros	<p>a) Learning process becomes faster by propagating rewards faster after <math>n</math> steps</p> <p>b) Propagating rewards to relevant state-action pairs that potentially make the algorithm efficient</p>
Cons	Explicitly compute n-steps returns that may increase the computational complexity
<b>Prioritised Experience Replay</b>	
Aim	It is an online reinforcement learning algorithm aims to provide faster and effective learning by using replay memory for experiences based on priority
Description	<p>Step 1: Initialise replay memory and store the experience <math>(s, a, t, r, s')</math> into it</p> <p>Step 2: Select actions from state and store transitions with the highest priority</p> <p>Step 3: Compute importance sampling and TD-error</p> <p>Step 4: Update transition priority and weights into the target network</p> <p>Step 5: Choose optimal action <math>A_t</math> from <math>\pi_\theta(S_t)</math></p>
Pros	<p>a) It requires less experiences to be trained due to prioritisation and can utilise more computation and memory</p> <p>b) Outperforms with sampling using Double DQN</p>
Cons	This can introduce bias for non-uniform sampling for experiences



<b>Asynchronous Advantage Actor-Critic (A3C)</b>	
Aim	A3C aims to provide a simple and robust solution in the domain of RL by using a global network and several sub-networks which performs asynchronously and update the global network.
Description	<p>Step 1: Initialise a global network and sub-network</p> <p>Step 2: Sub-networks interact with the environment</p> <p>Step 3: Compute value and policy-loss by the sub-networks</p> <p>Step 4: Sub-network gets a gradient from losses asynchronously</p> <p>Step 5: Sub-network updates global network with gradients after <math>t_{max}</math> actions or a terminal state is reached</p> <p>Step 6: After each update global network propagates new weights to the sub-network to share a common policy</p>
Pros	<p>a) This can be implemented for continuous and discrete action spaces</p> <p>b) It can perform better with the graphical processing unit (GPU).</p>
Cons	Employing too many agents or sub-networks can cause a computational delay which decreases the convergence speed.
<b>Actor-critic with experience replay (ACER)</b>	
Aim	ACER is an off-policy RL algorithm that aims to achieve stable and efficient learning by minimising the cost of simulation steps by using experience replay buffer.
Description	<p>Here, the algorithm is associated with a master algorithm that is responsible to call the associated algorithm for discrete actions.</p> <p>Step 1: Reset the gradients and initialise the parameters</p> <p>Step 2: Sample the trajectory from the replay memory</p> <p>Step 3: Compute the function for on-policy and quantities for trust region updating</p> <p>Step 4: Accumulate gradients and update the retrace target network asynchronously</p> <p>Step 5: Update the average policy network.</p>
Pros	<p>a) It can perform both in discrete and continuous domain spaces.</p> <p>b) It performs faster compare to standard actor-critic.</p>
Cons	Rare and infrequent experiences may lead bias.

## 2.9 Optimisation Techniques for DRL

Optimisation techniques are one of the key components in the study of deep reinforcement learning (DRL). Different optimisation techniques have transformed the field of AI (Badar, Umre and Junghare, 2014). In this section, the common ways of using various optimisers in the domain of DRL especially based

on gradient descent techniques have been discussed. This branch of DRL epitomizes a step toward building autonomous systems by understanding different experimental parameters and the aim to enhance their capacity to produce the best result in the current settings.

In other words, optimisers help to understand the RL agent to interact within the environment faster and effectively. It also helps to recognise the visual world slickly for a RL agent. Different optimisation techniques are mainly utilised to solve the complicated control problems that were previously obstinate and help to enlighten the DRL study. The common examples include learning to play video games directly from raw pixels, allowing control policies for robots to learn through trial-error directly from camera inputs in the real world (Li, 2017). In this section, the merits and demerits of the selected optimisers have also been mentioned.

### **2.9.1 Gradient descent**

A good optimiser can take a decision regarding which action is best and how to select the best policy in a deterministic or stochastic environment. Gradient descent (GD) is a common algorithm to perform optimisation of deep learning. It is the technique to minimise an objective function. Many deep learning libraries contain various gradient descent algorithms such as Keras, Chainer, Tensorflow, Theano and so on (Dan Clark, 2018).

These algorithms are regularly used as a black-box and users are often not clear what is going on inside these boxes. The following Figure 2.11 shows a typical visualisation of the optimisation technique based on the gradient descent where  $\nabla_{\theta}J(\theta)$  is the gradient with respect to the objective functions and parameters where it displays the local optima for a minimisation problem (Ruder, 2016).

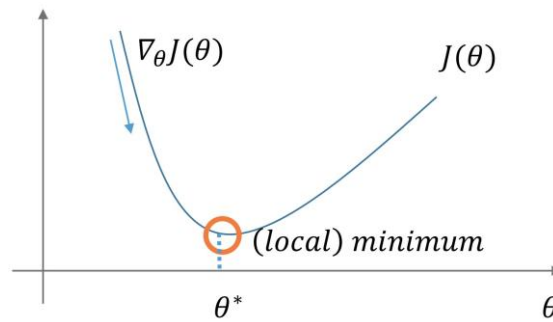


Figure 2. 11: Optimization with gradient descent (Ruder, 2016)

In a nutshell, adaptive learning rate methods such as Adagrad, Adadelta, RMSprop, Adam are particularly useful for sparse features concerning gradient descent approaches (Ruder, 2016; Le et al., 2011). In this study, Adam and stochastic gradient descent (SGD) optimisers have been used for the test case 1 and 2 respectively. Now, in the following section, these two optimisers have been reviewed.

### 2.9.2 Stochastic gradient descent (SGD) optimiser

SGD optimisers are normally much faster than gradient descent optimisers. In addition, this can be used to learn online (Ruder, 2016). In SGD, each parameter is updated in accordance with the minibatches and the learning rate  $\alpha$  which is smaller than the learning rate in batch gradient descent.

#### Pros:

- Much faster than batch gradient descent and allows online learning.

#### Cons:

- It causes high variance where the objective functions fluctuate frequently.

Keras Implementation of the SGD can be found in Appendix C.

### 2.9.3 Adam optimiser

Adaptive Moment Estimation (Adam) was introduced by (Kingma and Ba, 2014). This optimisation technique computes the learning rates for every parameter. Similar to other gradient descent algorithms, this optimisation technique keeps the average of the past gradients.

#### Pros:

- Adaptive learning rate and force for every parameter.
- Learning rate does not reduce as in Adaptive Gradient (AdaGrad).

#### Cons:

- Does not ‘look forward’ like Nesterov Accelerate Gradient (NAG).

Keras Implementation of Adam can be found in Appendix C.

## 2.10 Benchmarks for Multi-objective Optimisation

Different benchmarks have played an important role in analysing the algorithms for DMOPs. According to the literature, there are several approaches to deal with the dynamic multi-objective optimisation. In this study, a comprehensive review has been done in the context of different benchmarks especially in the evolutionary and in the RL settings. In the following section, a brief discussion has been conducted based on the evolutionary and the RL benchmarks.

### 2.10.1 Reviewing existing benchmarks for MORL

In this section, existing benchmarks have been reviewed which are used as a contextual study while designing a new benchmark in the MORL settings. For this reason, four benchmarks have been selected which appeared in most of the research works. The first one that has been analysed is the resource gathering

problem which is observed at first, by the work of (Barrett and Narayanan, 2008) for the strategy games. This is a 2D grid-world consist of 100 states with 25 grid cells. The agent can move one square among the four cardinal directions at a time. In this environment, the agent's task is to collect resources (i.e. gold and gems) and return to the home. An enemy has incorporated in this environment to attack the agent with 0.1 probability. When the agent is attacked by the enemy, it loses its current resources and must return to the initial location which is the home of the agent. Therefore, the reward vector is structured as (enemy, gold, gems). This benchmark represents a continuous environment. There are six non-dominated solutions according to (Barrett and Narayanan, 2008).

The second one is the PuddleWorld that is introduced by (Boyan and Moore, 1995) which was primarily used for a single-objective problem. It is a two-dimensional environment where the agent begins each episode randomly. Besides, the agent has to move to the top-right corner while avoiding the puddles. The agent moves to the left, right, up and down. While traversing the environment, the agent receives its current coordinates  $(x, y)$  as input. The obtained Pareto frontier is convex in this environment. Though, a number of local concavities are also observed by a closer inspection for PuddleWorld environment. However, (Vamplew et al., 2011) converted this single objective problem to multi-objective problem by presenting two penalties as separate components.

The third benchmark which has been analysed is the non-episodic Linked Rings where four deterministic policies exist (Vamplew et al., 2017a). There are four Pareto-optimal deterministic policies. This benchmark is a standard form of an MDP. Besides, it is used when there is no prior knowledge is available.

The multi-objective mountain car is the last problem that has been reviewed in this research work as a background study (Vamplew et al., 2011). Here, the agent drives the car where it needs to escape from a valley. While climbing, the car needs to obtain additional energy by reversing it. There are two objectives such as to minimise the number of steps and the second one is the minimising the number of

actions for left or right acceleration to reach the goal. The agent is penalised by -1 for every steps and acceleration action. Thus, this environment has got three objectives in total. Figure 2.12 shows all the reviewed environments in this study.

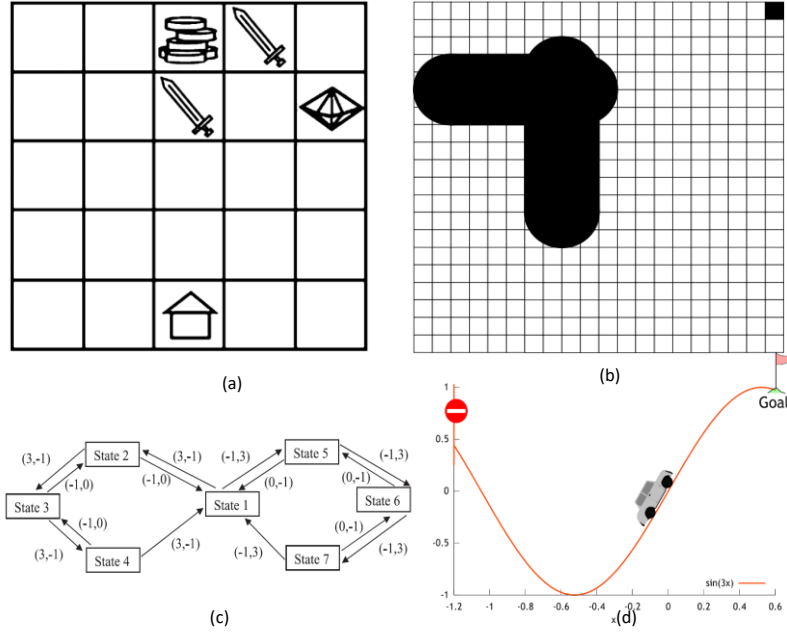


Figure 2. 12: Existing benchmarks for MORL (a) resource gathering (b) MO-Puddleworld (c) Linked Rings (d) Mountain Car

In addition to the above-mentioned benchmarks, there are some modified versions of the DST environment which have been studied in the literature. The Pressurized Bountiful Sea Treasure (PBST) Environment is among one of them which is proposed by (Moffaert and Nowé, 2014).

Similar to the DST testbed, PBST benchmark concerns with a deterministic episodic task where the agent is capable to go into one of the cardinal directions. Here, the agent aims to maximise the treasure value within minimum time as well as minimum water pressure. In other words, there are three objective functions where two of them are responsible for minimising the value and one is for maximising it.

Furthermore, some of the existing DRL testbeds have been analysed to get an overall understanding before designing and developing the proposed benchmark.

It has been observed that one of the popular testbeds in the context of the DRL is *OpenAI Gym* (Justin Francis, 2017). This is actually a toolkit developed in Python where the researchers are able to test, simulate and analyse different algorithms. Another common framework is the *Arcade Learning Environment* (ALE) to create and evaluate AI agents (Machado et al., 2017). This environment is consist of Atari 2600 games (Mnih et al., 2013). There are some other gaming environments for DRL research such as *StarCraft II* environment, *VizDoom* (i.e. AI-based environment for visual RL research) and *TORCS* (i.e. automobile racing simulator) (Vinyals et al., 2017).

Moreover, *twrl* is a framework for RL research by twitter (Torch for RL: Introducing torch-twrl, 2016). *RLGlue* (Tanner and Ca, 2009) is also a prominent Python-based library for RL research (Li, 2017). The following section is going to highlight the existing benchmarks based on evolutionary approaches.

### **2.10.2 Reviewing existing benchmarks for evolutionary approaches**

Evolutionary approaches have gained a steep ahead concentration in the literature to solve the multi-objective optimisation problem (Nedjah and Mourelle, 2015). The following list shows several approaches in brief that have been explored in the literature:

- a. Diversity-based approaches: In this approach, the solution space is adequately searched. The dynamic non-dominated sorting algorithm II (D-NSGA-II) (Deb, Rao N. and Karthik, 2007) and the dynamic constrained NSGA-II (DC-NSGA-II) (Azzouz, Bechikh and Ben Said, 2015) are prominent in this category.
- b. Change prediction-based approaches: This approach of solving DMOPs are useful when to exploit past information and anticipate the location of the new optimal solutions. Dynamic multi-objective evolutionary gradient search (D- MO-EGS) (Koo, Goh and Tan, 2010), dynamic multi-objective

EA with ADLM Model (DMOE/ADLM) (Li et al., 2014), the Kalman Filter Assisted MOEA/D-DE algorithm (MOEA/D-KF) (Muruganantham, Tan and Vadakkepat, 2016) are some of the examples in this approach.

- c. Memory-based approaches: This sort of approach uses extra memory to store useful information from the past generations to guide the future search. This technique can be utilised when the environment changes slightly such as the multi-strategy ensemble MOEA (MS-MOE) (Wang and Li, 2010), the adaptive population management-based dynamic NSGA-II (A-Dy-NSGA-II) (Azzouz, Bechikh and Said, 2017).

In order to evaluate the existing benchmark for the dynamic multi-objective optimisation, recently released benchmarks such as DF1 to DF14 are also analysed. These benchmarks are primarily used for the evolutionary algorithms (Jiang et al., 2018) and deal with up to 3 objectives.

In addition, some of the existing well-known evolutionary algorithms and their performances such as DTLZ2 and MOTSP are reviewed using PlatEMO as described in (Tian et al., 2017).

### **2.10.3 Generating benchmark for the dynamic MORL**

There are three approaches exist to design and develop the benchmark in the literature. These techniques are used to design the benchmark systematically (Li, 2016). The common approach is the multiple single objective function approach, constraint surface approach and the reverse engineering approach (Jiang and Yang, 2014; Fernandez, McCarthy and Rakotobe-Joel, 2001).

The first strategy is common while studying the multi-objective evolutionary algorithms. The main concern of this tactic is using different single objective functions to construct and solve multi-objective problems. The second and third



approach is concerned with the Pareto optimal front (Helbig and Engelbrecht, 2014a, 2014b, 2013b).

In this study, the reverse engineering approach has been utilised where the results need to formulate first such as setting up the true Pareto-front and then construct the benchmark corresponding to that. The reason for doing this is to provide the facility to test the different algorithms impartially (Li et al., 2008). This approach is also called the bottom-up approach to construct a benchmark.

Usually, in the case of reverse engineering, mathematical expressions and the construction of the environment are required to build a benchmark. Chapter 4 has a detailed discussion on these mathematical expressions as well as the definition of the MOMDP in the context of MORL settings for both test cases.

In the conclusion of the above discussion, a list of criteria has been set that need to be reflected while designing the benchmark. For example, the test problem selected for designing a benchmark must have the properties where an algorithm can be analysed and tested.

The followings points have been taken into consideration while designing the benchmark for the DMOPs in RL settings.

- The benchmark must be easy to construct and should not deviate a lot from the existing benchmarks.
- The benchmark must have the facilities to scaling up. In other words, it has to be robust.
- The benchmark should be capable to incorporate different variables and objectives so that any number of goals can be achieved based on the defined environment.
- Researchers should be able to incorporate different gameplay into the benchmark.

- The resulting Pareto frontier must be determined and easy to generate the non-dominated solutions.
- Corresponding decision variables must be simple to handle and easy to construct along with the other factors such as changing parameters and the constraints.
- The benchmark must be easy to converge to the true PF and it should allow multi-objectives possibly conflicting objectives.
- The benchmark should be able to incorporate the common performance measuring techniques to evaluate different algorithms
- Finally, the benchmark should be able to visualise the different PF and PS.

Considering the MOMDP approaches, the benchmark should have the facilities to accommodate the continuous or discrete search space whether the agent can traverse the environment fully or partially.

## **2.11 Reviewing the Metrics for Performance Evaluation**

When algorithms solve DMOPs, performance appraisal is in fact, vital to computing the proficiency of the algorithm. This is also useful to rank the different algorithms against each other. In the common setup for the multi-objective problem, the objective functions are conflicting to each other and the algorithm usually tries to find out non-dominated solutions. In this section, a brief review has been discussed in the context of MOO and DMOO problems.

### **2.11.1 Reviewing performance metrics for MOO problems**

In the area of multi-objective optimisation (MOO) problems, the agent tries to find out the set of solutions that are close to the true PF where the solutions are diverse and evenly spread along with POF. However, the decision maker then decides the optimal solutions once POF is found based on the requirements and preferences.

An outperformance relation has been introduced by (Hansen, Hansen and Jaszkiewicz, 1998) under the following assumptions which is taken into consideration in this study.

- The preferences are unknown a priori to the decision taker
- Let the  $PF_A$  and  $PF_B$  are the two solutions, the decision maker will consider  $PF_A$  because of its outperformance iff  $PF_A$  is better for a specific preference and not worse in any other attributes provided by the  $PF_B$ .
- The preferences can be modelled in terms of functions.

Like the performance metrics in the evolutionary computational area, especially for the MOO, there are also metric in order to efficiently evaluate and compare the performance of an algorithm in a dynamic environment. There are five main performance characteristics to evaluate an algorithm in the context of the dynamic optimisation environment when finding an approximation of the true PF, they are as follows (Mirjalili and Lewis, 2015):

- convergence,
- distribution,
- the number of the obtained Pareto optimal solutions,
- track the moving global optima, and
- not stuck in the local optima.

The first characteristic is responsible to the convergence of an algorithm towards the true PF. In this scenario, the goal of the agent is to find out the close and accurate approximation of the robust and optimal PF. After that, the algorithm is responsible in terms of the performance measure that shows the capability of finding evenly spread robust and optimal solutions regarding PF. In addition, the algorithm is judged based on the number of found vigorous and non-robust optimal PF solutions. In this thesis, the considered algorithms are effective if and only if a particular algorithm is capable to find out the robust optimal PF solutions as much as possible and avoid finding the PF solutions which are not robust. In other words,

the success of the algorithm is considered if it can find the solutions in the robust region in the dynamic multi-objective optimisation environment. The fourth characteristic is dealing with the tracking of the optima in the dynamic environment. It is obvious that in the dynamic environment, the optima will be moving from one place to another (Morales-Enciso and Branke, 2014).

Therefore, the agent has to trace these moving optima. Finally, in the settings of the RL, the agent often gets stuck in the local optima because of the greedy approach by the agent while exploring the states in the environment (Andersen, Goodwin and Granmo, 2018). Therefore, it is also essential that the algorithm is capable of having a healthy balance for exploration and exploitation for traversing and visiting all the nodes in a reasonable timeframe (Hazrati, Hamzeh and Hashemi, 2013).

### 2.11.2 Reviewing performance metrics used in DMO algorithms

It is frequently argued that the agent does not truly solve the dynamic MOOPs, yet, through a procedure of knowledge is gained by interacting within the environment and training over time, the agent starts to behave in harmony with the optimal solution (Başçö and Orhan, 2000).

The following Table 2.6 shows the performance metrics that have been studied to analyse the algorithms.

Table 2. 6: Performance metrics and their uses in analysing DMO algorithms

Types	Performance metric	Referenced in
Diversity	Maximum spread (MS) metric shows a higher value of MS that reflects the coverage of optimal PF by the obtained PF.	MS is introduced in (Chi-Keong Goh and Kay Chen Tan, 2009) and used in (Azzouz, Bechikh and Ben Said, 2015; Azzouz, Bechikh and Said, 2017)
	Path length (PL) measure is used to calculate the distance between neighbouring solutions on the optimal PF.	PL is introduced in (Mehnen, Wagner and Rudolph, 2006) Used: nil
	The coverage scope (CS) is used to enumerate the coverage of the set of non-dominated solutions.	CS is introduced in (Zhang and Qian, 2011) Used: nil

Accuracy	Generational Distance (GD) is used to quantify the distance between the optimal PF and the true PF	GD is introduced in (Mehnen, Wagner and Rudolph, 2006) and utilised in (Chen, Li and Chen, 2009; Amato and Farina, 2005)
	Inverted generational distance metric (IGD) specifies the distance between the optimal PF and the evolved PF.	IGD is introduced in (Sierra and Coello Coello, 2005) and utilised in (Azzouz, Bechikh and Ben Said, 2015; Aimin Zhou, Yaochu Jin and Qingfu Zhang, 2014; Muruganatham, Tan and Vadakkepat, 2016; Wang and Li, 2010, 2009; Azzouz, Bechikh and Ben Said, 2014)
	Variable space generational distance metric (VD) is used to calculate the closeness between the approximated PF and the optimal one.	VD is introduced in (Chi-Keong Goh and Kay Chen Tan, 2009) and used in (Koo, Goh and Tan, 2010; Azzouz, Bechikh and Ben Said, 2014)
	Success ratio (SC) measures the ratio of the obtained results which are the members of the true PF.	SC is introduced in (Mehnen, Wagner and Rudolph, 2006) Used: nil
Robustness	Stability counts the consequence of the changing location on accuracy.	Stability measure is first introduced in (Weicker, 2002) and utilised in (Camara, Ortega and Toro, 2007)
	Reactivity measures the capability of the algorithm to react to variations by evaluating the time to attain an anticipated goal.	Reactivity measure is introduced in (Camara, Ortega and Toro, 2007)
Combined measure	Hypervolume (HV) reflects a space dominated region in the objective space.	HV is introduced in (Zitzler and Thiele, 1999) and utilised in (Camara, Ortega and Toro, 2007; Wang and Li, 2010; Zheng, 2007; Liu, Wang and Ren, 2015)
	Hypervolume (HV <sub>r</sub> ) ratio determines the dominated objectives in the objective space.	HV ratio is introduced in (Veldhuizen, Van Veldhuizen and Van Veldhuizen, 1999) and used in (Deb, Rao N. and Karthik, 2007; Azzouz, Bechikh and Ben Said, 2015; Aberdeen, Thiébaux and Zhang, 2004; Deb, 2011)

In addition, the following Table 2.7 shows a list of widely used methods for the statistical tests in the dynamic multi-objective optimisation sector.

Table 2. 7: Performance metrics and their use in analysing DMO algorithms for the statistical tests

Category	Statistical assessment	Referenced in
Parametric	t-test	Utilised in (Aimin Zhou, Yaochu Jin and Qingfu Zhang, 2014; Wang and Li, 2010)
Non-parametric	Wilcoxon test	Utilised in (Azzouz, Bechikh and Ben Said, 2015; Azzouz, Bechikh and Said, 2017; Azzouz, Bechikh and Ben Said, 2014)
	Kolmogorov-Smirnov test	Used in (Koo, Goh and Tan, 2010; Chi-Keong Goh and Kay Chen Tan, 2009)

Among them, three evaluation metrics are used to assess the effectiveness of the proposed algorithm. This includes generational distance measure (GD), inverted generational distance (IGD), Hypervolume (HV) and parametric tests which are described in Chapter 6. The following section describes the applied part of the research.

## **2.12 Existing Process of Water Quality Evaluation in São Paulo, Brazil**

As discussed in the literature, many studies are focused on the academic problems where the objective functions are assumed, and constraints are pre-populated and investigated. However, it is not clear whether these problems are still related to real-world problems or not.

In this thesis, to make the selected problem relevant and connected to the real-world scenario, the dataset (Publicações e Relatórios | Águas Interiores, 2017) of water quality in São Paulo, Brazil has been utilised to identify and predict the vulnerable zones based on water quality resilience. In this section, the existing process of water quality evaluation in the targeted area has been discussed. The frequency of data collection is different considering various data collection points. In this study, our targeted area is in São Paulo that is located in the southeast of Brazil ( $23^{\circ}33'S$  and  $46^{\circ}38'W$ ) and its total municipal area is of 1,521.11 km<sup>2</sup> (see Figure 2.13). Companhia Ambiental do Estado de São Paulo (CETESB, Brazil, 2000) has been monitoring the quality of surface water in the state of São Paulo since 1974. The main purposes of monitoring are:- i) to make a diagnosis of the quality of the state's surface waters, assessing their compliance with environmental legislation; ii) to assess the temporal evolution of the quality of the state's surface waters; and iii) to identify priority areas for the control of water pollution, such as stretches of rivers and estuaries. The assessment of freshwater quality is complemented by temporal and spatial analysis. To set up the trends of WQI, the dataset from the previous 17 years (2000-2016) are considered. The space was carried out by drawing up health profiles of the main water bodies in order to identify critical sections. The presentation of water quality index (IQA),

quality of water for public supply purposes (IAP), protection of Marine Life (IVA), trophic state (IET), bathing (IB) and Biological Communities (Phyto and zooplankton and Benthic organisms represented by ICF, ICZ and ICB, respectively), are also part of the evaluation.

According to CONAMA Resolution No. 357 of March 17, 2005 (CONAMA Resolution 357/05 – Brazilian NR, 2005), the sweet surface waters are classified according to the quality required for their main uses in five quality classes.

In the existing process, the systematic evaluation of the water quality by CETESB has the following objectives.

- Needs to keep up with population growth,
- Diversification of industries in the state,
- Control programs of water pollution developed by CETESB and
- Diagnosis of the sources used for public supply.

Table 2.8 shows the existing practices by CETESB from 1974 where the ‘Points’ column refers to the added stations from the corresponding year.

Table 2. 8: Freshwater monitoring networks in São Paulo State - 2017

Monitored Sources	Goals	Start of Operation	Points	Frequency	variables
Basic network	Provide a general diagnosis of water resources in the State of São Paulo.	1974	461	Bimonthly	Physical Chemical Biological
Sediment Network	Complement the diagnosis of the water column.	2002	26	Yearly	Physical Chemical Biological
Bathing Rivers and reservoirs	Inform water conditions for primary contact recreation / bath to the population.	1994	35	Weekly / Monthly	Biological
Monitoring Automatic	Control of domestic and industrial pollution sources and control the quality of water for public supply.	1998	12	Hourly	Physical Chemical

For each of the uses and quality grades, were established by means of variables such as unnatural floating materials, oils and greases provide substances which taste or smell, colouring matter from anthropogenic sources, toxicity and

objectionable waste. In terms of the quantitative measures, the existing practice contains the collection of pH, BOD, OD, organic substances, and total dissolved metals, cyanobacteria density, chlorophyll data. The maximum allowable limit of the variables for each class of water is called quality standards. Figure 2.13 illustrates the targeted area (São Paulo State) for this study.

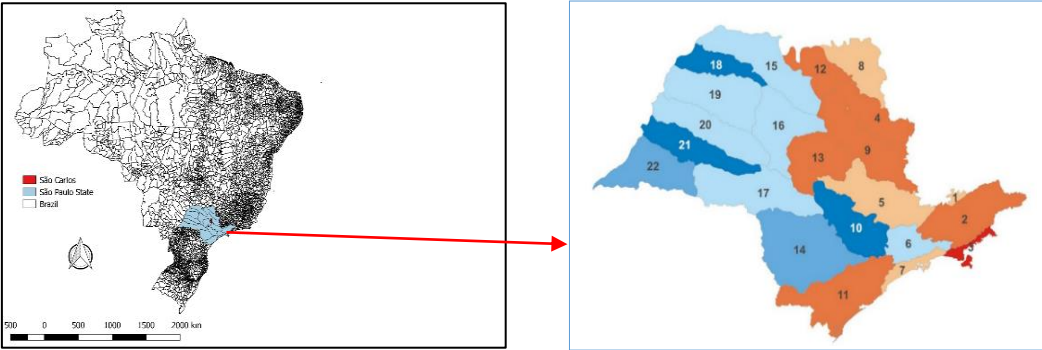


Figure 2. 13: Targeted area (São Paulo) to implement the proposed framework

The distribution of the monitored points in terms of different types of economic area is depicted in the following Figure 2.14.

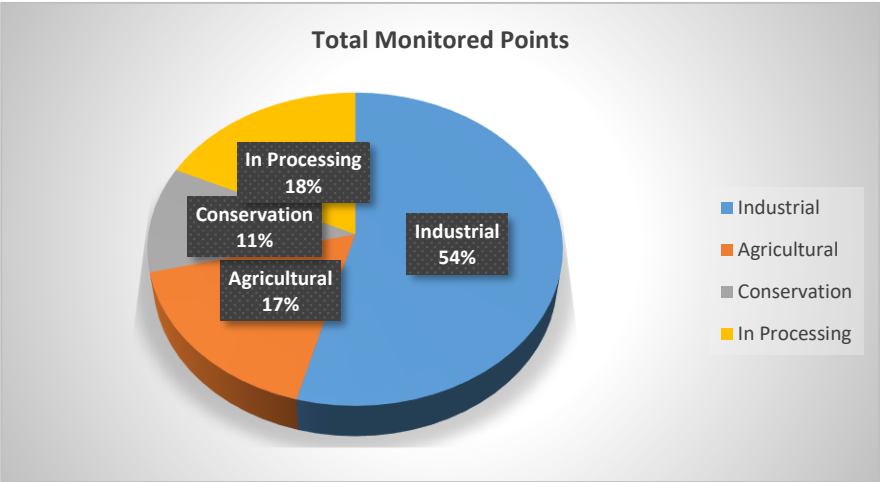


Figure 2. 14: Distribution of sampling points based on different types

As discussed earlier, the factors of water quality resilience are changing throughout the year due to various reasons and thus, the quality variables are changing. Hence, this problem has formed an appropriate real-world test case in a dynamic multi-objective environment.



### **2.13 Reviewing Machine Learning Studies for Water Quality Evaluation**

The objective of this section is to utilise the concept of Machine Learning (ML) and its usefulness for test case 2. Here, the implementations of ML for water quality evaluation have been reviewed. The water resources degradation and pollution particularly in developing countries are major concerns due to their unsustainable developments (Seto, Güneralp and Hutya, 2012; FN and MF, 2017; Lu et al., 2015; Paul, 2015). Therefore, ML suitability to evaluate the water quality can have a fruitful impact and effective approach in the existing process.

Water quality has a strong impact on the natural environment and human life; therefore, it is a primary concern in the case of surface water and reservoir management. In recent years, dry areas have been facing severe water shortage, which is an alarming issue for the environment (Li et al., 2017). Recent studies reveal that about 5.5 billion people will face water crisis in 10 years (Amitrano et al., 2014). Besides, the disposal of sewage in rivers and lakes are adding more to this threat to arid areas (Lindberg et al., 2014; Zhou et al., 2014). The volume and quality of waters (Lima, Lombardo and Magaña Rueda, 2018) in reservoirs (e.g. rivers, lakes etc.) are very important not only for the environment but also for societal and economic development (Pawara et al., 2017).

Hence, ensuring water quality is one of the crucial conditions of the overall development. An AI-based solution can enhance the whole process more easily and cost-effectively. Reservoirs not only provide pure water for human consumption but also water for various other purposes, like agriculture, industry and habitats for aquatic lives (Hoverman and Johnson, 2012). Various properties of water in reservoirs, especially its quality, must be assessed. Assessing the quality of water critically enables managers to develop optimal water resources management plans.

In recent years, a resilience-based approach has been extensively promoted and used in various aspects of the water systems management, but the potential is not equally explored and developed. Resilience has a wide range of definitions and it can vary from one context to another. Resilience-informed WQ management (Imani et al., 2016) will take a holistic approach and focuses on the promotion of key drivers, attributes and role players' adaptive capacities to cope with the changing conditions rather than the use of control-based risk management (Etchepare and van der Hoek, 2015) or treating the effects. Therefore, a resilient socio-ecological system (SES) can handle negative man-made and environmental changes without regime shifts and by improving and using the system adaptive capacity more effectively. A resilient WQ management provides a framework to view a system that continues operating when subject to challenges over time and space.

Moreover, WQ spatial and temporal changes have been widely studied and used to identify and assess WQ characteristics due to natural and man-made influences (Hendry, Gotanda and Svensson, 2017) or for life cycle assessment of geomorphological dynamics (Tooth and Stephen, 2018). In the latest studies, a spatiotemporal variation of resilience has been used to evaluate watershed health (Sadeghi and Hazbavi, 2017).

WQ monitoring approaches are generally expensive and time-consuming, particularly for large-scale and complex catchments (Zeng et al., 2013). Also, there are several deterministic and stochastic WQ models (SWAT, RMT, WMS etc.) available to manage the best practices for WQ management (Einax et al., 1999; Hull, Parrella and Falcucci, 2008). However, most of these models are very complex and demand a significant amount of field data with post data collection analysis. Furthermore, many statistical-based WQ models, make assumptions for simplification, which can negatively affect the accuracy and reliability of the analysis (Chen and Liu, 2015). Therefore, utilising statistical approaches does not necessarily hold high precision (Najah et al., 2013).

It is inexorable that artificial intelligence (AI)-enabled system can be impactful in environmental sectors by capturing non-linear and complex relationships among the influential factors more precisely and intelligently than the traditional or manual methods. AIs are often used for prediction purposes through the development of data-driven models. For example, artificial neural networks (ANN) has been extensively used for prediction and forecasting in water systems studies (Elsafi, 2014; Güldal et al., 2010), especially river WQ prediction (Khalil et al., 2014; Palani, Liong and Tkalich, 2008; Matthews, Hilles and Pelletier, 2002; Sarkar and Pandey, 2015) and WQ indexes (Gazzaz et al., 2012; Elshemy and Meon, 2016). In addition, there are many advantages of using ANNs for prediction purposes such as: eliminating the need to a priori knowledge the underlying process and the existing complex relationships of the system elements (Kalin et al., 2010).

Additionally, a wide range of water quality indexes in relation to different water bodies has been used in many conventional studies using ML techniques. For example, Chou, Ho and Hoang (2018) used four ML techniques such as artificial neural network (ANN), support vector machines (SVM), classification and regression trees (RT), and linear regression (LR) to analyse the quality of water.

Moreover, there are several studies based on multiple linear regression methods combined with AI to develop WQ models (Slaughter et al., 2017; Tomas, Čurlin and Marić, 2017). An AI system was developed by (Ji et al., 2017), combining multiple models based on SVM, ANNs, LR to prove the supremacy of SVM in predicting dissolved oxygen (DO) concentration in Wen-Rui Tang River, China. Similarly, to predict the level of DO, a general regression neural network (GRNN) was proposed by (Antanasijević et al., 2014) for the Danube River, Europe. ML algorithms and remote sensing spectral indices have been used to evaluate the water quality by (Wang, Zhang and Ding, 2017). A predictive model is developed using long and short-term memory neural network (LSTM NN) by (Wang et al., 2017).

The application of deep learning (DL) has some degree of success in WQ evaluation as discussed in background studies. A deep learning method is also used for mapping the surface water (Isikdogan, Bovik and Passalacqua, 2017). In this study, DRL-based WQ resilience evaluation model has been used as a surrogate of complex computational WQ models which demand a great deal of data for a preliminary evaluation of surface water resilience in the case study area. Additionally, this surrogate approach will save time and cost (e.g. financial, computational, etc.) of data collection and monitoring used for WQ resilience evaluation.

However, reviewing the latest literature, hardly any studies have been found that measure water quality “resilience” using artificial intelligence. In this study, deep reinforcement learning technique has been used to predict the vulnerable zones based on water quality resilience. In this study, three separated networks for each water quality index (WQI) have been produced. A detailed discussion of formalising MOMDP using DRL for WQ evaluation has been conducted in Chapter 4.

## **2.14 Justification of the Study**

Literature shows that no significant works have been done for the dynamic multi-objective environment in DRL settings. As a result, to the best of the author’s knowledge, there is no benchmark in the dynamic (Farina, Deb and Amato, 2004) multi-objective DRL context. Therefore, this study has addressed this gap by applying the existing knowledge to propose a benchmark which may help to investigate in the simulated environment (i.e. test-case 1) as follows:

- a. Understanding the dynamics while objectives are conflicting with each other and
- b. Applying the existing knowledge to deal with the constraints and problem parameters that change over time.

In addition, in the context of water quality resilience, there has been no use of state-of-the-art machine learning techniques. Though a few studies have been conducted in hydrodynamics (French et al., 2017) and other fields such as ocean engineering (Sarkar et al., 2015), there is hardly any research done in resilience-based approaches for water quality evaluation (Mugume et al., 2015). As discussed earlier, to investigate the applicability of a well-known deep learning algorithm, researchers cannot ignore the dynamic behaviour in the multi-objective environment in RL settings. Therefore, in this study, this behaviour has been incorporated and discovered the way to fit the dynamics in the multi-objective environment.

Moreover, the concept of our proposed benchmark has been applied to address a real-world problem. In addition, an algorithm has been developed that can handle dynamics in the multi-objective environment based on DQN. To prove the concept with an empirical result, the proposed algorithm is designed to tackle the dynamics and its complexity in the simulated environment. In these environments, the proposed algorithm is integrated to examine whether it can satisfy the goal. Another aspect of this algorithm is to identify and predict the vulnerable zones based on water quality resilience to observe whether the deep RL domain can contribute in the domain of hydrodynamics or decision support systems and to direct the researchers for further investigation.

In a nutshell, a successful implementation of an AI-enabled system may have a significant impact on test case 2:

- a. To categorise the impacts of water contamination on public supply in terms of resilience,
- b. Minimising the manual efforts to collect the data from different zones and
- c. Identifying the vulnerable zones that need prioritisation in the decision-making process for necessary interventions to make the diagnosis process faster while preserves the accuracy.

## 2.15 Summary

In this chapter, several concepts have been discussed which are necessary to follow the overall thesis. At first, the intelligent application, dynamic environment and the MORL approaches are discussed. Later on, the basics of RL and DRL have been discussed. Furthermore, various challenges for studying DMOPs are also evaluated. Here, a review of the DMOPs has been described that is normally used by the researchers to examine their algorithms in various settings.

Furthermore, a brief discussion of the main features and distinction of the current optimisation techniques are put forward. After that, different dynamic benchmark problems and the real-world DMOPs have been explained in the context of evolutionary as well as RL approaches. This chapter also outlined the necessary considerations while generating test problems and the benchmark. These were useful to assess and compare the considered multi-policy RL algorithms for the DMOPs which have been utilised in Chapter 5 and 6.

With the review of the recent RL success, it can be concluded that DRL is one the promising techniques to train the machine to respond to a changing environment and then take evolutionary actions based on the policy. In this chapter, an argument on the different approaches has been described to solve the RL problem and came up with the two prominent solutions such as single and multi-policy RL approaches.

Furthermore, a brief discussion of the different ways of solving MORL using DRL techniques has been explained. Moreover, different DQNs and their aim, advantages and disadvantages have been described which are necessary to build the proposed algorithm as it is based on DQNs. Two optimisers which have been used in this study, are elaborated and their merits and demerits are identified. Likewise, several difficulties and scopes were also analysed that still need to be handled while using DRL. Figure 2.15 illustrates the sequence of reaching the corresponding methods which are going to be applied in this thesis.

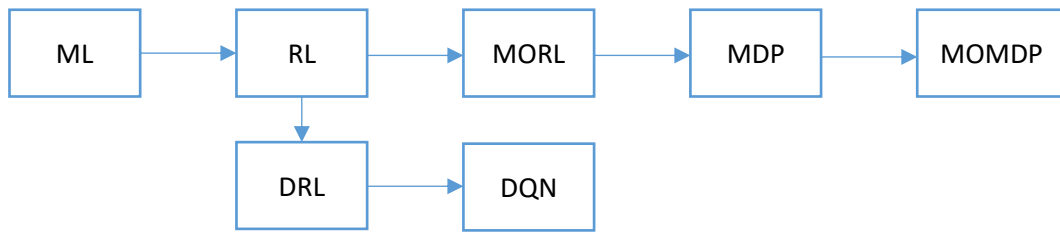


Figure 2. 15: A flow diagram to select the necessary methods

Furthermore, the existing process of water quality evaluation has been described in the state of São Paulo, Brazil. A critical review has been done to identify the existing flaws in the current procedure of the targeted area. After that, different ML techniques have been reviewed which have been used for water quality study. Finally, this chapter is concluded with the justification of the study. Methodology of this study will be discussed in the following chapter.

## Chapter 3

### Methodology

#### 3.1 Introduction

There are good numbers of researchers who believe that RL is one of the best technologies for creating general AI (Scarcello, 2019; Artificial intelligence: The saviour of mankind or the end of the world?, 2018). It is an interesting field that has huge potential to solve many intractable problems. The recent success of DRL has opened the door of many opportunities. However, getting started with DRL requires some initial tasks to be performed which are discussed below.

The main challenge in this research work is to formalise the MOMDP considering the simulated environment both for test case 1 and 2. Therefore, the methodology to conduct this research work is set carefully that requires several procedures which are going to be discussed in this section. Method details for both test cases will also be discussed in this chapter. Moreover, data collection, preparation, and how resilience has been calculated will also be explained in this chapter. The steps are as follows:

- a) At first, a comprehensive review of the existing solutions has been conducted. During reviewing the literature, recent advances in DMOPs have been studied and a knowledge gap has been identified. Thus, the test cases have been finalised in terms of the benchmark and the real-world scenario. Therefore, the first objective of this research is going to be satisfied.
- b) Secondly, a mathematical model and a computational model have been designed and developed which helps to form a new and innovative DMOP testbed. Through this process, a wide-ranging requirement elicitation has been conducted and a prototype is created for the benchmark. Thus, the second and third objective will be achieved.



- c) After that, a novel method for multi-objective optimisation algorithm in the dynamic environment has been established. In addition, the proposed algorithm has been evaluated to improve its efficiency. An advanced machine learning algorithm such as DRL has been used to build this proposed algorithm backed by DQN architecture that provides an appropriate mapping in dynamic MO constrained environments. Thus, the fourth objective is going to be accomplished to reach the aim of this thesis.
- d) Finally, a real-world problem has been addressed to predict the vulnerable zones based on the selected parameters in one of the populous cities in Brazil. Hence, the final objective of this research is going to be obtained.

### **3.2 Research design**

To achieve the research goal and objectives, an indicative research method and overall approach are being discussed and illustrated below. In this study, the following key factors are being considered to roll out the research.

- a) To gain profound knowledge and understanding of the decision support systems, an extensive literature review has been conducted from different sources such as top-quality journals, conferences, recent news, blogs and articles.
- b) After analysing recent research works in this domain, a conceptual model in the context of dynamic multi-objective decision support system has been developed.
- c) Necessary and relevant algorithms have been analysed and evaluated to form a new and advanced algorithm.
- d) To prove the concept, a real-world test case has been considered to predict the critical zones based on water quality resilience.
- e) Finally, the results and outcome of this research have been evaluated.

In the high-level abstraction, the overall methodology of this study was divided into four working packages (WPs). The first package is responsible to provide the

necessary algorithms by analysing the existing algorithms from the literature. In this package, the research gap has been identified. Moreover, a widespread reviewing of the existing testbeds has been conducted to form the benchmark in this package in the context of a multi-objective dynamic environment.

Formalising the MOMDP for the DMOPs is handled by the second package. This package 2 is responsible for designing and developing the proposed algorithm. Training of the model and continuous improvement is also an integral part of this package.

The third package is responsible to evaluate the proposed algorithm and the benchmark. Moreover, in this package, the trained model has been integrated into the test cases. In this package, the evaluation procedure is also examined for both test cases and finalised the algorithm.

The final package is responsible for incrementally improving the whole study so that the obtained outcome can have significant scientific and knowledge contributions. This package also complies with the performance test module to update the overall progress and performance according to the actual results. It also gives the final output through the empirical analysis by conducting a critical evaluation. Figure 3.1 shows all the working packages (WPs) as per the above discussion.

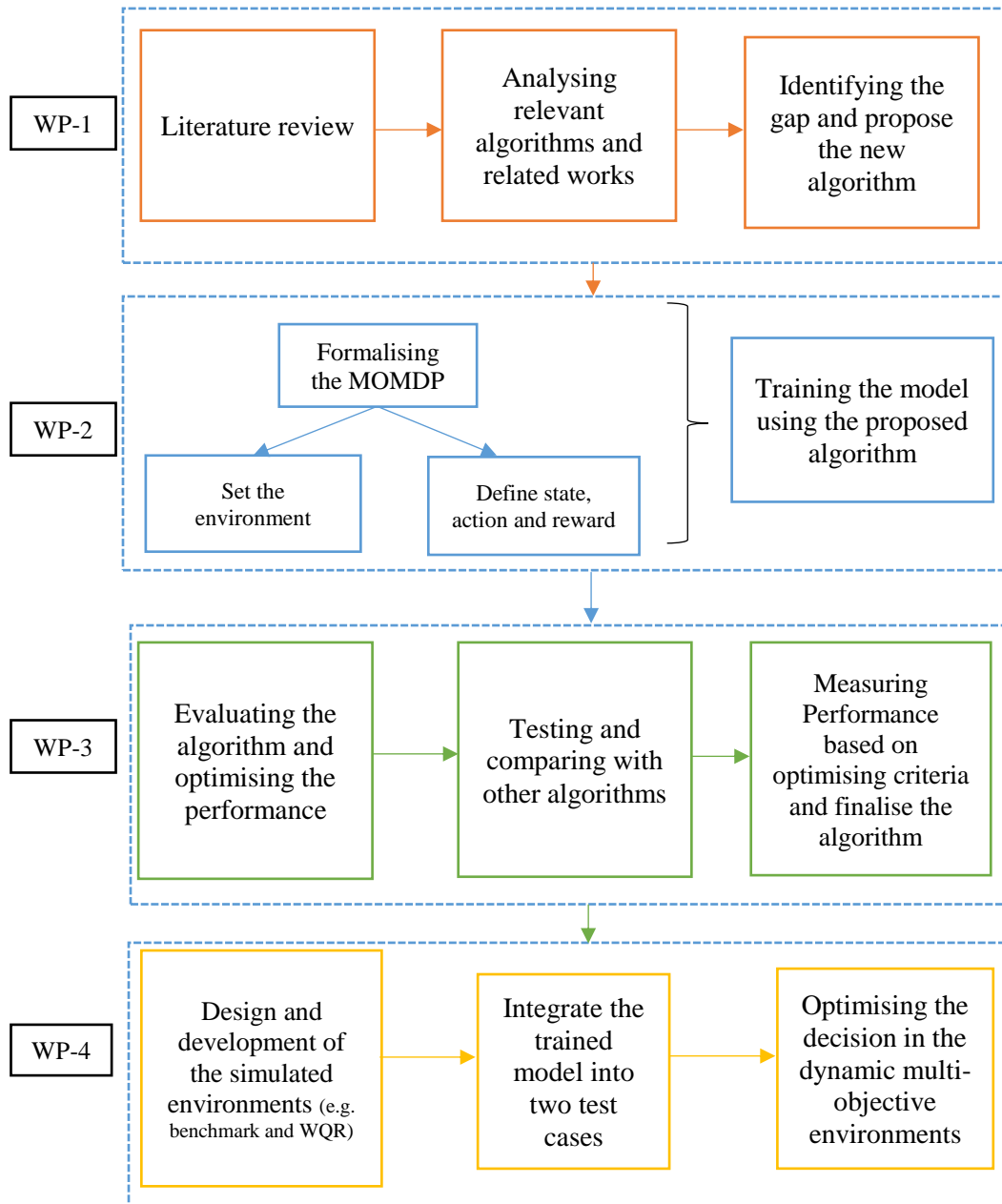


Figure 3. 1: Working packages for this research work

As discussed earlier, a virtual environment has been considered which satisfies the required components of this research such as a multi-objective decision-making scheme and optimising the decisions based on that scheme. The following diagram 3.2 shows an overview of the system architecture that has been followed while designing the research methodology for this study.

There are four modules in the overall system architecture as shown in Figure 3.2. The first module is the system input or in other words, where the end user interacts with the system. The second module is the controller between the end user and the proposed algorithm. In this module, the system performs through an application programming interface (API) that works with the Matlab and the Python library.

The data stream is responsible for the data fusion from the two different sources where the agent gains reward based on the observed state and actions which helps to determine the meta-policy selection. This reward is based on the vector rewards and thus, the optimal policy is selected. This policy determines which action needs to be sent through the API for further process. The final module is the core part where the algorithm performs an overall action based on the MOMDP and the DQN.

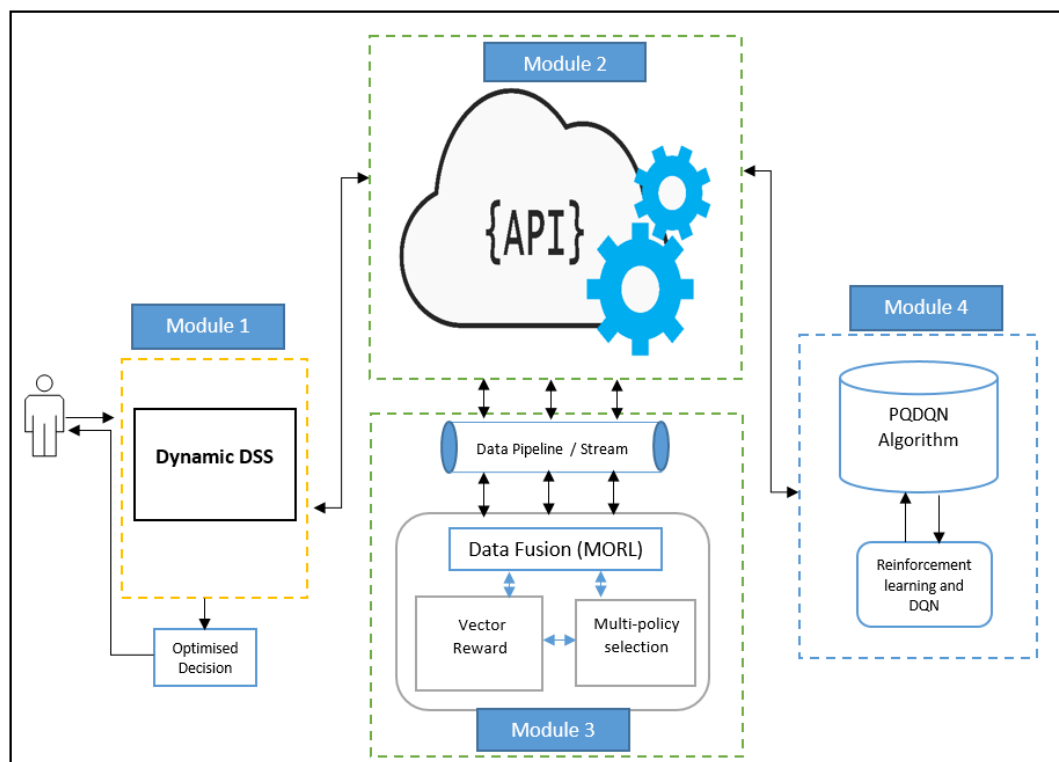


Figure 3. 2: Overview of the system architecture

### 3.3 Approaches to defining the conceptual model

The testing of any algorithm in the DMOPs is usually complex considering the dynamics of a problem that involves a wide range of different scenarios including uncertainty. Besides, to conduct an empirical study, the approach to solve the problem needs to be defined precisely as stated by (Cruz, González and Pelta, 2011). Therefore, a detailed plan of the defined approaches to the experiment has been exemplified in this section.

To define the problem, a conceptual model has been sculpted as DMOPs to generalise the objective functions in an abstraction level where some elements of the underlying model changeover the course of the optimisation.

In general, the following Equation 3.1 is considered as the DMOP based on MOMDP:

$$DMOP = \begin{cases} \text{optimise } f(x, t) \\ \text{s.t. } x \in F(t) \subseteq S, t \in \text{finite time} \end{cases} \dots\dots (3.1)$$

Where  $S$  is the search space,  $t$  is the time,  $f: S \times T \in \mathbb{R}$ , is the objective function, and  $F(t)$  represents the feasible solutions at time  $t$ . In other words, the MODOP in this context is considered as a dynamic problem where the objective function or the restrictions are changed over time.

To solve this problem, the simplest solution would be ignoring the dynamics and solving the problem by dividing the multi-objective into a single objective problem. The next step is to scalarise the reward to achieve the highest expected reward in RL settings.

However, this will not satisfy the goal of this thesis and will not reach any pragmatic contribution. As a result, the goal of this experimental problem settings

is not only directed to locate the stationary optimal solution but also to track the changing optima in the dynamic environment. Therefore, before setting up the environment for the RL, a protocol needs to be defined that restricts the setup components.

As discussed earlier, the agent is nothing but the algorithm that interacts within the environment. This environment can be deterministic, discrete or continuous and stationary or non-stationary. However, in this study, the environment is considered based on the following certain factors:

- a. The agent should adapt to the environment which is a 2D grid-world,
- b. The environment must be observable (e.g. fully or partially),
- c. The environment or area should be traversable by a single agent and
- d. The agent should receive vector rewards instead of scalar rewards.

When the agent interacts with the RL environment, the state model is either deterministic or non-deterministic and this can be represented by deterministic finite automata (DFA) and non-deterministic finite automata (NDFA) (Rathod, Marathe and Vidhate, 2014). As stated earlier, the agent needs to deal with the vector rewards where the actions are not pre-defined and fixed. Thus, the selected actions that lead to forming the policy is determined based on the current state and the associated action for that state.

Therefore, the agent interacts within the NDFA where the actions are depended on the current state and the associated vector rewards for a particular action. In addition, the pre-defined environment is needed to be observed whether it is wholly or partially observable. For instance, a chess game is a fully observable whereas the Poker game is partially observable such as the cards are unknown of the one's player. Besides, only a single agent environment is considered in this study.

In a single agent-based environment, an agent only interacts with one environment. However, it is to be noted that the proposed single agent interacts in a single

environment which is dynamic such as the parameters and objectives are changed over time.

Figure 3.3 visualises a typical setup for the single agent environment in the context of the DMOPs for RL settings where the agent (i.e. triangle) lives in the environment (i.e. oval). The agent's task is to update its goal based on the actions and accumulated knowledge.

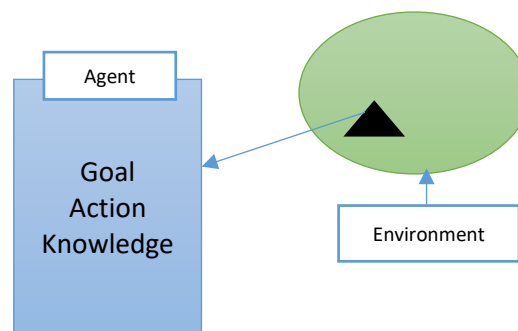


Figure 3. 3: An agent traversing within a single environment

### 3.4 Method Details for Test Case 1

In this section, the method details for test case 1 is discussed. The selection process of test case 1 was not straightforward. The first task was to analyse the existing benchmark and then find out the appropriate benchmark that can be robust and satisfy multi-objective preferably conflicting objectives.

The following Figure 3.4 illustrates test case 1 with the reward distribution. As mentioned earlier in Chapter 1, the agent's objective is to find out all the state vectors for a problem where these define a non-convex Pareto frontier in a reward space. Figure 3.4 also shows the traditional DST environment that demonstrates the treasure values with the true Pareto front.

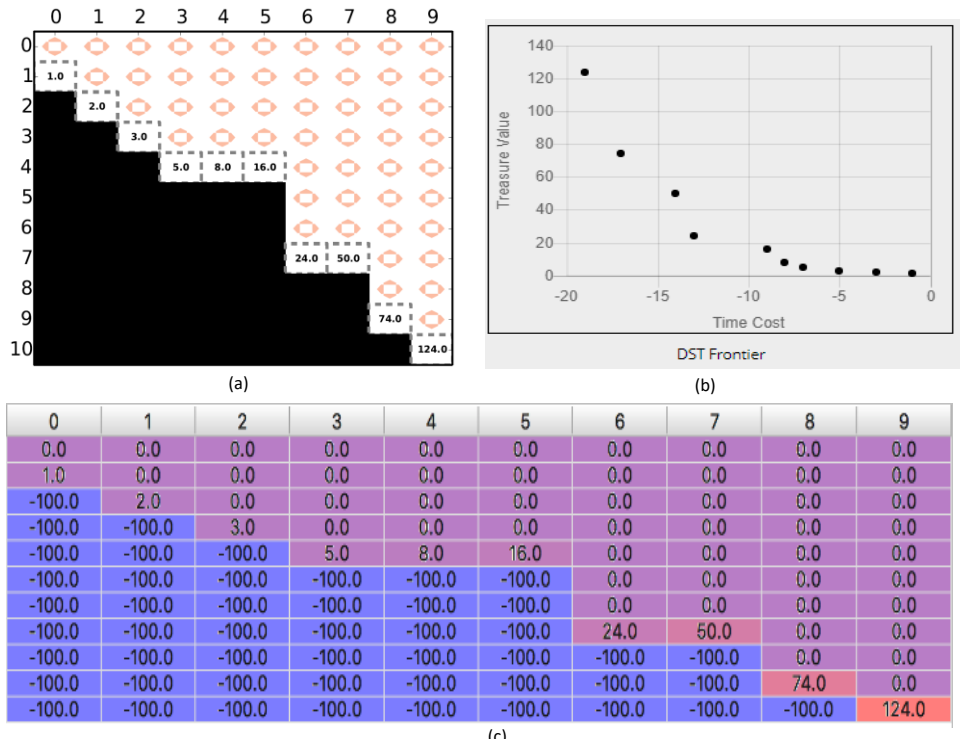


Figure 3. 4: Traditional Deep-sea Treasure problem (a) environment (b) frontier (c) reward distribution over the grid (i.e. extracted using Python)

To solve the RL problem, two approaches can be followed to formalise and solve the problem. The first approach is taking the raw image as input (Mnih et al., 2015) while the second approach is forming an MOMDP by hardcoding (GitHub - ttajmayer/morl-dv, 2018; GitHub - RL-LDV-TUM/morlbench, 2018).

### 3.4.1 Raw-image approach

The DRL research community has observed how the raw image input can be useful to learn 49 games by an agent with only raw frames with  $210 \times 160$  pixels and 128-colour palette (Mnih et al., 2015). In this process, the computational time and the usage of the memory could be highly expensive. Though a basic pre-processing can reduce the input dimensionality (Wilson, 2018). Usually, this procedure helps to encode a single frame to convert and points out the particular coordinates (x, y) of the current location of the agent or the treasure value. Figure 3.5 shows a possible visualisation of this process.



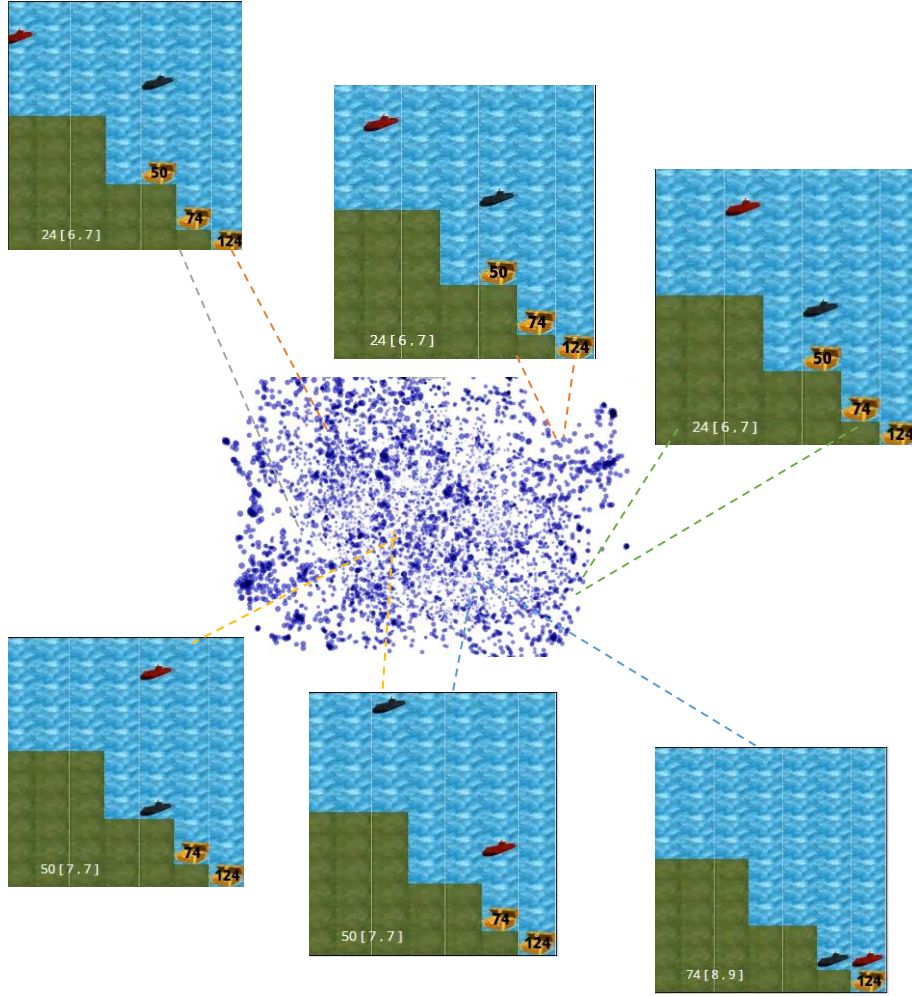


Figure 3. 5: Raw image approach to solve the test case 1 (DST environments)

On the other hand, the easier and computationally less complex approach with lower memory usage is to formalise the MOMDP using hardcoding. This includes the matrix functions to form the required components such as the grid-world of  $10 \times 09$ , state, action and reward functions. The following sections highlight the appropriate procedure that has been followed in this thesis.

### 3.4.2 Hardcode approach

Considering the hardcode approach, the DST treasure hunt is a straightforward game which needs to be coded whereas the agent has to find out the highest treasure values with minimum time, and for every step, the agent has to be penalised by -1 point. At first, it needs to be considered that the agent in the grid-

world is moving around in a finite horizon. In other sense, this world represents an environment using a controlled MDP where the agent works as a controller in that MDP.

The methodology for setting up test case 1 can be categorised with the following five key points.

### I. Defining the model:

At first, the MDP needs to be set up which defines the environment. In this step, the multi-objective Markov decision process (MOMDP) needs to be defined as a sample shown in the following Figure 3.6. Here, the agent's task is to move from  $S_0$  to  $S_1$  where objectives are defined by  $O_1$  and  $O_2$  and the vector reward of setting up the model with the appropriate actions  $a_0$  and  $a_1$ . Here,  $D$  represents a dynamic value that helps to balance the objectives. The model is set based on the vector rewards and punishments. Thus, the agent receives the information of a state whether it is a current state or not and which action needs to be taken based on the reward. By learning this mechanism, the agent moves and takes its action as like as the joystick movement. After that, the environment reacts to this action and runs the next state associated with the reward function.

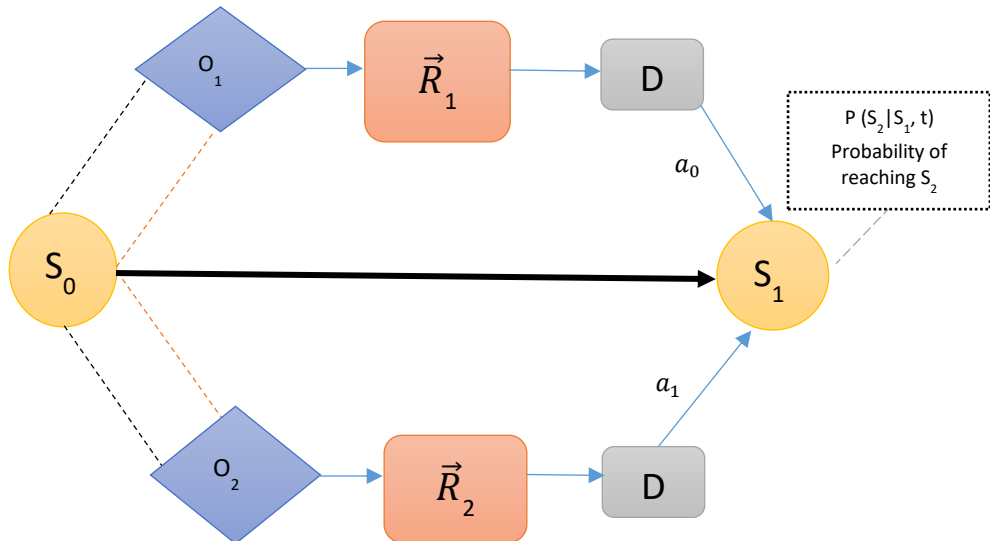


Figure 3. 6: A simplified visualisation of the MOMDP

The rules that have been used for choosing the action is called a policy. The above-mentioned environment is generally stochastic where the next state is not known by the agent. If only 1 episode is considered as mentioned in the above Figure 3.6, in a finite sequence of states, actions and rewards, then the formation of the tuple will be as like as the following Equation 3.2:

$$S_0, a_0, r_1, S_1, a_1, r_2, \dots S_{n-1}, a_{n-1}, r_n, S_n \dots \dots \dots (3.2)$$

Where,  $a_0$  and  $a_1$  represent the actions and the episode ends with the terminal state  $S_n$ .

## II. Selecting future discounted rewards:

In this process, a discounted reward function needs to be defined which will help the agent to perform better in the long run. So that the agent not only looks at the current benefit by looking for only the immediate reward but also attempts to maximise the discounted future rewards. To do so, the total vector rewards need to be calculated for one run in an MOMDP. Since the environment is stochastic, which means the agent will not be sure that it is going to get the same rewards in the next state by performing the same actions. The more time the agent spends, the more the agent may diverge. Therefore, the solution has to use the discounted future reward as shown in Equation 3.3.

$$\vec{R}_t = \vec{r}_t + \gamma \vec{r}_{t+1} + \gamma^2 \vec{r}_{t+2} + \dots + \gamma^{n-t} \vec{r}_n \dots \dots \dots (3.3)$$

Here  $\gamma$  represents the discounted future reward that ranges between 0 and 1. This range means that as long as the agent depends on the future, it values less to the present situation. Therefore, the optimised choice for the agent would be selecting an action that maximises the future reward.

### III. Perception for the Q learning:

In Q-learning, a function is defined as  $Q(s, a)$  representing the maximum discounted future reward while the agent makes an action  $a$  in state  $S$  at time  $t$  until the agent finds the optimal Q value from that point. In a simplified way, the Q value function can be defined as the following Equation 3.4.

$$Q(s_t, a_t) = \max(\vec{R}_{t+1}) \dots \dots \dots (3.4)$$

Where the agent maximises the total vector reward  $\vec{R}$  at time  $t + 1$ . The other important task is to define and select the DQN network which is discussed in Chapter 5 systematically.

### IV. Training process:

At this point, an idea is recognised that how to use the future reward in each state using the Q-learning function. There are still needs to employ some other tactics for the purpose of the convergence. The agent has to store the experiences that are observed through the interaction within the environment. These experiences are comprised of the sequence of episodes. Hence, the replay memory needs to be utilised because of the approximation of Q-values using non-linear functions which is not a good choice for stability.

Therefore, the experience replay needs to be deployed. Throughout the interaction, all the experiences  $\langle s, a, r, s' \rangle$  are kept in a replay memory as mentioned in Chapter 2. This process benefits the agent to take random minibatches instead of the recent transition while training the network. The other benefit of doing so is that it helps the agent not to be stuck in a local minima for random sampling.

Let us consider a buffer with a batch of experiences, the agent needs to find out the possible actions by predicting the future reward  $Q(s', a')$ . After that, the agent

determines the  $\max Q(S', A')$  and train the network using a gradient loss function. This training determines how the network value is closed to the target value.

## V. Exploration-exploitation dilemma:

The final part to set the model is to create the exploration mechanism for Q learning. Sometimes, the agent chooses the best option can end up with some unvisited nodes because of a Q network or table is initialised randomly and the predictions are also generated randomly. Therefore, always selecting the best option might not be the solution to this scenario. In terms of the Q function converges, the agent returns most consistent Q-values, as a result, the exploration decreases. To avoid this dilemma, an  $\varepsilon - greedy$  policy has been utilised instead of greedy policy (e.g. may be stuck in local optima without searching all the nodes). Therefore, this process ensures that the agent can traverse all the nodes including the unexplored one.

In a nutshell, this method includes  $n^{\text{th}}$  episodes where the agent receives the probabilistic rewards by observing the current state, selecting and performing an action. It also involves an immediate payoff and updates the target network to maximise the highest expected reward.

### 3.5 Method Details for Test Case 2

Since the last century, many representations of intrinsic data structure and approaches to learning patterns have been proposed such as linear or nonlinear, supervised or unsupervised, shallow or deep. Particularly, deep architectures are widely applied in recent years and have produced top results in many areas including image classification, speech recognition, anomaly detection and so on (LeCun, Bengio and Hinton, 2015). In this test case, one of the cutting-edge techniques has been applied such as deep reinforcement learning (DRL) to identify and predict the vulnerable zones in the state of São Paulo, Brazil.

DRL is a well-known method for self-learning and has successfully been implemented in various sectors such as playing games, image processing, power optimisation and so on to achieve human-level expertise (Mnih et al., 2015). However, it is still ambiguous whether DRL can be implemented for a historical dataset in a dynamic multi-objective optimisation problem by formalising a dynamic MOMDP. In this section, this procedure has been explored which is responsible to build the model for water quality resilience that results in detecting critical stations and incorporated in a real-world scenario using RL techniques. Here, an easy and fast procedure has been produced to predict the resilience from a historical dataset (2000-2015), produced by Companhia Ambiental do Estado de São Paulo (i.e. CETESB, the Brazilian authority for monitoring the quality of the surface water) in a multi-objective environment using deep reinforcement learning (Governo Do Estado De São Paulo, 2018). In a nutshell, this method is significant because of the followings:

- a. This method can significantly reduce the manual efforts which are primarily expensive, time-consuming and slow to perform.
- b. This process can be used on any water quality dataset to predict the resilience based on the set of parameters on the local standardisation of the surface water.
- c. A wide variety of different analysis and techniques can be anticipated which may add value to water quality studies.

As this test case 2 is based on the multi-criteria and the changing parameters, therefore, it is essential that this study provides a substantial amount of concentration in the multi-criteria system. In the context of multi-criteria decision analysis, the following stages are considered (Multi-criteria analysis: a manual, 2009):

- a) Establishing the decision context,
- b) Identifying alternatives,
- c) Identifying criteria and sub-criteria,
- d) Analysing weight of each factor and sub-factor,

- e) Finding the performance of each criterion and
- f) Deciding the best alternatives as a decision in a dynamic environment.


To plan for the improvement of WQ resilience, it is necessary to identify and prioritise the critical zones in the case study region for interventions. However, this is a challenging task in this research due to the conflicting interactions amongst WQIs. Hence, this study has developed a method using outcomes of the predictive model using WQ resilience model to identify the critical zones (i.e. from the most critical to the least one) in the case study region.

A multi-criteria ranking environment has been adopted by using a decision matrix as mentioned in (Hasan et al., 2016) for the given WQI datasets as presented in Table 3.1. This matrix uses a bottom-up approach to reach a particular decision where every criterion is dependent upon its associated sub-criterion.

In this study, the parameters in Table 3.1 are defined as:

- Selected decisions ( $D_i$ ): critical zones
- Alternative decisions ( $E_i$ ): critical stations in each zone
- Criteria: WQI i.e. IQA, IET and IVA
- Factor: monthly values of WQI
- Sub-factor: WQIs' resilience values in each WQ monitoring station in each zone
- $z$ : number of zones

Table 3. 1: Multi-criteria decision matrix

Selected Decisions ( $D_1, \dots, D_z$ )					
Alternative Decisions ( $E_1, \dots, E_n$ )					
 Bottom-up approach	Factor 1	WQI <sub>1</sub>	.....	WQI <sub>n</sub>	
	Factor 2	WQI <sub>2</sub>	.....	WQI <sub>n</sub>	
	Factor 2	WQI <sub>3</sub>	.....	WQI <sub>n</sub>	
		⋮	⋮	⋮	⋮
	Sub-factors (SubF <sub>1</sub> )	WQR <sub>1</sub>	.....	WQR <sub>n</sub>	
	Sub-factors (SubF <sub>2</sub> )	WQR <sub>2</sub>	.....	WQR <sub>n</sub>	
	Sub-factors (SubF <sub>3</sub> )	WQR <sub>3</sub>	.....	WQR <sub>n</sub>	

The bottom-up approach is formed to satisfy the multi-criteria for selecting the vulnerable zones. Here, the process has been categorised into three phases.

Phase 1: To determine the factors as mentioned in Table 3.1, corresponding WQI values have been associated in this phase with all the monitored stations.

Phase 2: In this stage, to determine the weights of each resilient station, a weight has linked to each WQ station for all the 22 zones.

Phase 3: In this phase, the resilience values of WQIs of all the WQ monitoring stations (in all 22 zones) are calculated and linked with the stations.

The following Figure 3.7 demonstrates the scale of significance of each criterion that ranges from 0 to 1.

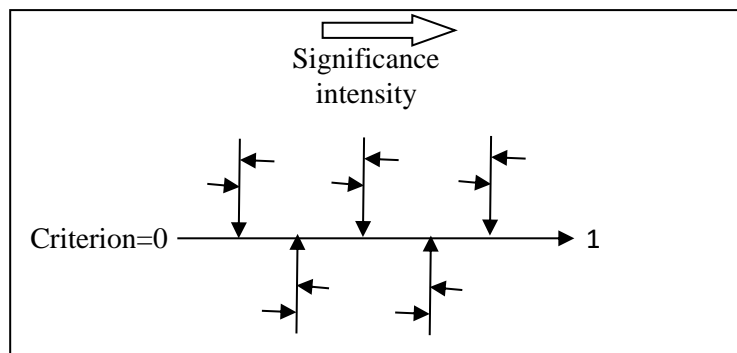


Figure 3. 7: Sub-criteria impact on each criterion

This scale helps to set the weights for the different stations in the 22 zones in the WQ dataset. The weights are used based on the triangular fuzzy number and calculated based on the pair-wise matrix. This scale is also adopted from (Hasan et al., 2016; Rao et al., 2017).

### 3.5.1 Data collection and preparation

A recent survey among data scientists by CrowedFlower (i.e. a data mining and crowdsourcing company based in the USA) shows that data preparation is the most time-consuming task which takes 80% time of the whole job (Gil Press, 2016). This primary and fundamental task is known as data wrangling or data munging (Endel and Piringer, 2015). It also refers to gathering, extracting, cleaning and



storing of data. Transforming data into a useful form remains challenging and tedious (Endel and Piringer, 2015). However, in this study, the following procedures of data wrangling are considered:

- a) Gathering data from CETESB (e.g. raw data which is in text formats in 16 PDFs and then converted into *csv* and *json* formats by using Python and pandas library (Python Data Analysis Library, 2018))
- b) Cleaning data (e.g. to check the completeness and accuracy by using python and pandas library)
- c) Identifying missing data (e.g. using *dropna* and *fillna* functions in pandas library)
- d) Measuring data quality (e.g. validity, consistency, uniformity by manually random check as well as using *pydqc* (SauceCat, 2017))

In 2017, CETESB's core network has operated in 62 automatic stations and 27 manual monitoring points in São Paulo (see Figure 3.8 and Table 3.2). The basic network of freshwater has 461 sampling points distributed by major rivers and reservoirs and 12 automatic monitoring stations. The network assessment of groundwater quality has 313 points. The integrated monitoring network quality and quantity have been expanded to 38 points, installed in the main state aquifers. The network of coastal waters consisted of 66 monitoring points in estuaries and the Atlantic Ocean.

It is worth mentioning that the monitoring of these networks is being done by CETESB for more than 40 years (Publicações e Relatórios | Águas Interiores, 2017). The constraints in the current procedure include a huge human resource allocation and financial load. Figure 3.8 shows the water resources of Brazil by cubic meters per capita.

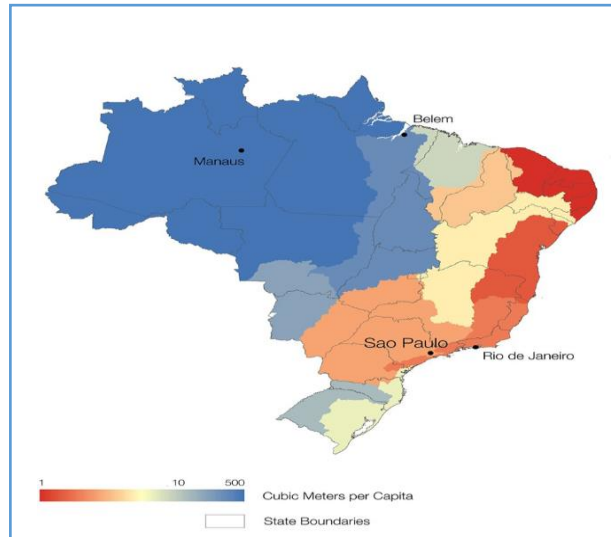


Figure 3. 8: Water resources of Brazil by Basin (cubic meters per capita) [Source: Brazilian National Water Agency, ANA]

### 3.5.2 Water quality parameter selection and resilience calculation

At this point, a comprehensive study has been conducted to select the parameters for calculating the resilience for test case 2. To formalise the MOMDP, it was essential to set these parameters to predict water quality resilience.

The first step was to select the parameters based on WQ dataset. To do so, an intensive analysis of the parameters has been conducted. The selection of quality variables is determined by the type of monitoring networks such as the basic network, bathing, sediment and automatic. In this thesis, the selection of variables is dependent on the usages of water checked by CETESB in partnership with the Department of Water and Energy of the State of São Paulo (DAEE). CETESB determines about 60 water quality variables (e.g. physical, chemical, hidrobiológicas, microbiological and ecotoxicological) considered the most significant (see Table 3.2). These variables are determined by at least 70% of the basic network. In the case of measurements, the rules to determine the flow rate are often periodically adjusted.

Table 3. 2: The quality variables of the Basic Network (freshwater)

Monitoring Network	Quality Index	Main purpose	Network Points	Variables that make up the index
Basic network	IQA	effluent dilution(Mainly domestic)	All	Temperature, pH, dissolved oxygen, biochemical oxygen demand, Escherichia coli / coliform thermotolerant, Total Nitrogen, total phosphorus, total solids and turbidity.
	IAP	Public supply	Utilised for public supply	Temperature, pH, dissolved oxygen,Biochemical Oxygen Demand, Escherichia coli, Total Nitrogen, Total Phosphorus, Total Solids, Turbidity, Iron,Manganese, Aluminum, Copper, Zinc, Trihalomethanes formation potential, number of cyanobacteria cells (lentic Environment), Cadmium, Lead, Chromium Total, Mercury and Nickel.
	IET	eutrophication	All except the rivers	Chlorophyll a and Total Phosphorus.
	IVA	Protection of aquatic life	classified in Class 4 (CONAMA 357/05) presenting bad quality	Dissolved oxygen, pH, Ceriodaphnia dubia ecotoxicological Assay, copper, zinc, lead, chromium, mercury, nickel, cadmium, surfactants, Chlorophyll a and Total Phosphorus.
	ICF	Protection of aquatic life	lentic environments used to supply; or state mesotrophic	Phytoplankton community, phosphorus and chlorophyll a
	ICZ	Protection of aquatic life	some reservoirs	Community zooplankton and chlorophyll a
Network bathing	IB	Bathing / Recreation	All	Coliforms or thermotolerant Escherichia coli or Enterococci
Sediment Network	CQS	Protection of aquatic life	All	Chemical contaminants that have values set by the ECCM (1999); Ecotoxicological test with Aztec Hyalella, Benthic Community
	ICB	Protection of aquatic life	Points that do not have bad quality / very poor in water	Benthic Community

In this study, the basic network and its IQA (Índice de Qualidade das Águas), IET (Índice do Estado Trófico) and IVA (Aqua Índice de Vida Aquática) as shown in Table 3.2 have been considered. IQA mainly refers to the contamination of the water bodies caused by domestic sewage. IET reflects the quality of water for nutrient enrichment and its impact on the growth of algae and cyanobacteria. It is also calculated on a priority basis for the protection of aquatic life. IVA is used to assess the quality of water for the protection of the aquatic life, including its essential variables for aquatic organisms, toxic substances and degree of trophic.

Threshold values for resilience evaluation of the selected parameters are shown in the following figures 3.9 and 3.10.

The following Figure 3.9 shows the IVA and the IQA parameters where threshold values are classified into 5 categories starting from the high quality to the very poor quality.

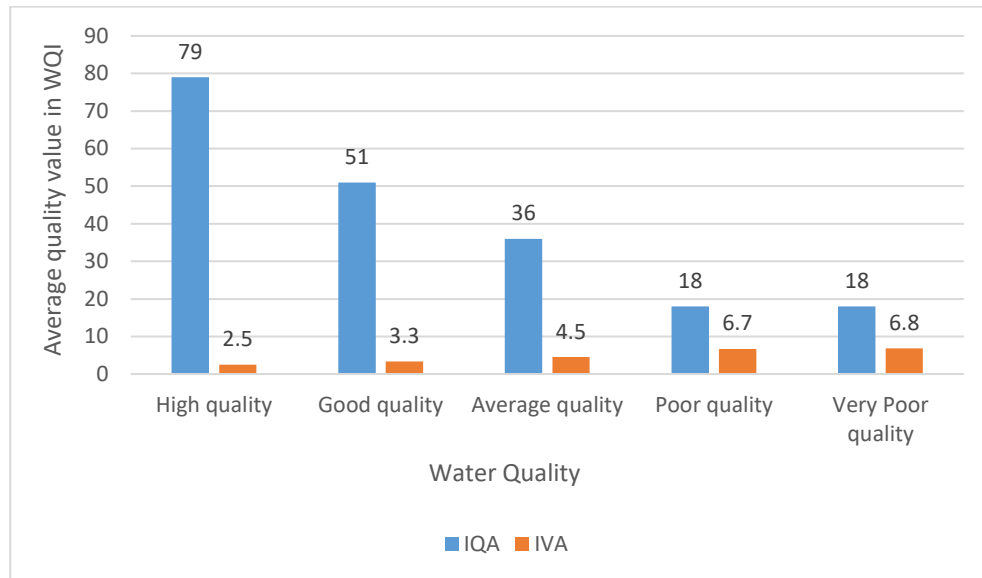


Figure 3. 9: Threshold values for IVA and IQA based on the water quality parameters in São Paulo, Brazil

On the other hand, as mentioned earlier, in the case of IET (as shown in Figure 3.10) the lower values are the better and it has been classified into six categories starting from the high oligotrophic to very highly eutrophic. See Appendix B for the threshold values in a numeric form.

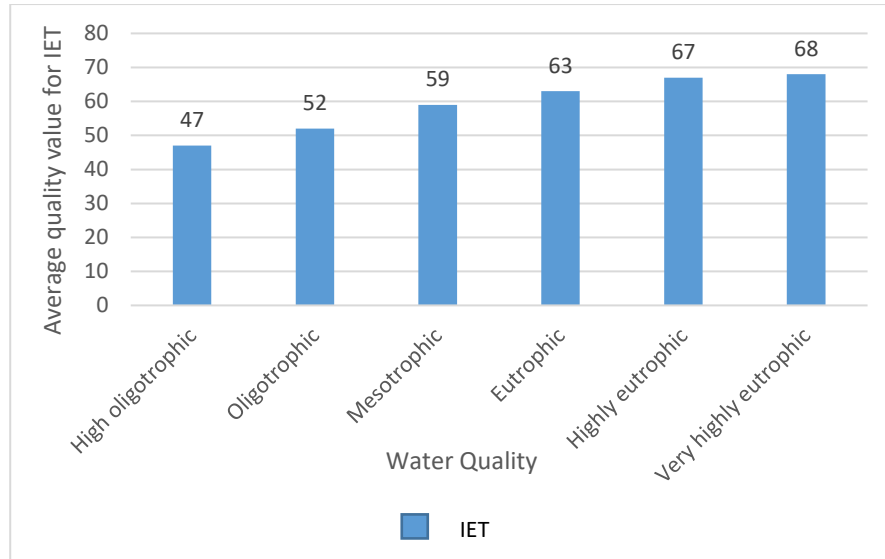


Figure 3. 10: Threshold values for IET on the water quality parameters in São Paulo

These above two Figures 3.9 and 3.10 show the conflicting behaviour of the selected parameters and thus, form an ideal conflicting objective scenario in the context of MOMDP in a dynamic RL setting.

To identify an appropriate methodology for test case 2 in a real-world scenario, the threshold values of the WQ parameters and resilience need to be calculated. Therefore, resilience calculation in the context of predicting WQR has been described to validate the applicability and competency of the proposed framework.

For resilience evaluation, the lower bound of average quality (i.e. IQA=36, IVA=3.4 and IET=52) for each water quality index has been considered as the threshold in resilience evaluation. This could be justified by the fact that the surface water quality, within the average quality range still has maintained essential quality factors.

Resilience is calculated based on Equation 3.5.

$$R_i = 1 - \int_{t_1}^{t_2} (F_{max} - F_i(t))(t_i - t_1). dt \dots \dots \dots (3.5)$$

where,  $R_i$  is the resilience at the time  $T_i$ ,  $F_{max}$  is the maximum water quality index,  $F_i$  is the water quality index at the time  $T_i$  and,  $t_i$  is the time elapsed and  $t_1$  is the

failure start time. In this study, resilience is scaled between 0 and 1 and hence, different water quality variables could be compared. One of the most used scaling methods i.e. Min-Max, as shown in Equation 3.6, is used for this purpose.

$$R_i^n = (R_i - R_{min}) / (R_{max} - R_{min}) \dots\dots\dots (3.6)$$

Where,  $R_i^n$  is the normalised value,  $R_{min}$  and  $R_{max}$  are the minimum and maximum values for each water quality index, respectively. It should be noted that the worst and best water quality indexes for IQA and IVA are interpreted in a different way than IET. The higher IQA indicates a better quality of surface waters while this is opposite regards to IET and IVA (i.e. the lower is better). Therefore, the lowest value ( $R_{min}$ ) is set to 0 and the highest value ( $R_{max}$ ) is set to the maximum resilience value over the period of resilience evaluation.

As like as the DST testbed, this test case also provides the dynamics where the values are not static over time. These data are changing which depends on natural factors such as weather, temperature, drought, precipitation and man-made changes such as usages and contamination caused by human (Lima, Lombardo and Magaña, 2018).

Finally, in this step critical zones are determined using Equation 3.7. For this purpose, WQ monitoring stations are ranked based on their total weight. The higher value of total weight represents more criticality and the lower ones represent less criticality. Here, the critical stations have been selected based on the IET, IQA and IVA as mentioned earlier. The threshold values have been set for the selected parameters as mentioned earlier. This includes the highest values for IET with the lowest values of IQA and IVA for a station. Moreover, a zone has only considered if it has more than two critical stations.

Drawing on that, Equation 3.7 presents the alternatives (i.e. critical stations) in the case study area.

$$Z = [S_z^{cs}] \dots \dots \dots (3.7)$$

where,  $Z$  denotes the critical zones across the case study area (i.e. 22 zones);  $z$  is the number of zones; and  $cs$  denotes the number of critical stations in each zone.

### 3.6 Summary

This methodology is based on empirical research and can be replicated to solve similar problems. It is worth stating that deep learning in the context of supervised and unsupervised learning has received widespread adoption in the AI and ML community. However, when it comes to DRL it has some ambiguity. In this chapter, the possible approach to form a RL agent has been illustrated. In addition, the method details for the purpose of building an MOMDP are also explained comprehensively.

In a nutshell, the following Figure 3.11 and the queries can be helpful to get started or replicated the whole process for defining the RL model for both test cases.

- I. How can the RL model be formalised using Markov Decision Process with long-term strategies?
- II. How can the future reward be approximated?
- III. How can the deep Q network be selected?
- IV. How to set the strategy for exploration and exploitation for the Q learning algorithm along with experience replay technique?

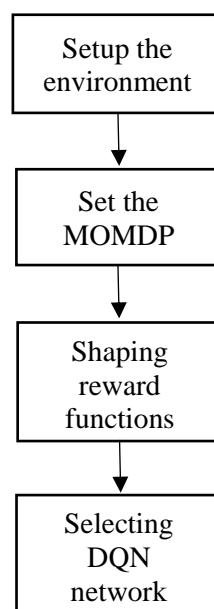


Figure 3. 11: Steps to form the RL agent in this study

Moreover, the proposed DRL based framework is utilised to predict the critical zones effectively that can be integrated with the conventional approach (Governor Do Estado De São Paulo, 2018) which may lead to a cost-effective solution. The efficient method can leverage multi-level controls to detect the water quality resilience in each station.

Furthermore, DRL algorithms may be effective to investigate any unexplored complexity compared to the traditional approach. Moreover, this model can be utilised for other water catchments, reservoirs or rivers (subject to the local standard of the water quality indices). The initial phase of the project can be considered of a quantitative nature to build the model. The model uses numerical input data that is gathered from the CETESB dataset for water quality in 22 zones in São Paulo, Brazil.

In this chapter, a guideline is provided to form an MOMDP, selection of the water quality parameters and the necessary procedures for calculating the resilience. This starting point may help further to determine the critical or resilient zones and the current forecasting practices and constraints. Figure 3.12 shows a glimpse of how to repeat the process where the novelty is to formalise the MOMDP for a historical dataset in RL settings.

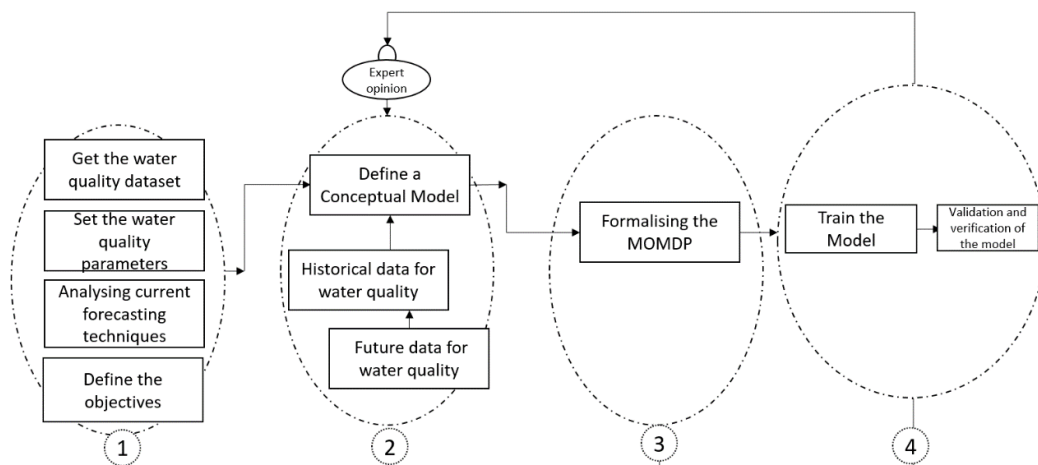


Figure 3. 12: Process to repeat the work in a different dataset



To sum up, this study can help the concerned authorities to check the water resilience level. The manual approach of data collection is expensive, time-consuming and difficult in certain cases due to adverse weather conditions. The proposed method may eradicate these problems.

Since the optimisation and the RL technique require lots of mathematical expressions and it is at the heart of any machine learning algorithm, the followings were necessary to form the equations and build the algorithm:

- Linear algebra
- Probability and statistics
- Calculus
- Optimisation procedures

The next chapter will be dealing with the problem settings and the experimental setups based on the methodology that has been discussed here.

## Chapter 4

### Problem Settings and Experimental Setups

#### 4.1 Introduction

Recently, significant advancements have been achieved in the field of DMOPs (Bechikh, Datta and Gupta, 2018). However, only a few studies are related to defining the problem and classifying them. To define the dynamics in the multi-objective environment, Farina et al. (Farina, Deb and Amato, 2004) classified the dynamic problems based on the changing Pareto Front (PF) and Pareto Set (PS). This classification also shows the difficulty of solving the DMOPs by describing the changing PF and PS. However, it does not deal with the changing objectives, parameters, and time-linkage properties (Nguyen, Yang and Bonsall, 2012). In addition, this classification also does not look at the source of the dynamics of the problems (Helbig, Deb and Engelbrecht, 2016). Therefore, benchmark problems and constructing the problem are still an important and challenging task in the context of DMOPs (Helbig and Engelbrecht, 2013).

On the other hand, algorithms that solve DMOPs should be tested on the benchmark functions to check the capabilities of a particular algorithm to overcome the difficulties in a specific point and how it is close to the true PF (Younes, Calamai and Basir, 2005). In this chapter, at first, the problem and the overall dynamics of the defined problem have been formulated. After that, the mathematical model of the defined problems has been articulated where the objective functions are generated based on the decision variables and the constraints. Moreover, one of the contributions of this thesis is represented in this chapter which is creating a DMOP benchmark in the context of RL settings. Besides this, experimental setups for both test cases have been discussed that is followed holistically to test the performance of all the considered algorithms.

## 4.2 Defining the Dynamic Multi-objective Optimisation Problem (DMOP)

A dynamic multi-objective optimisation problem (DMOP) can be defined as a vector of decision variables  $x(t)$ , that satisfies the objectives and constraints that change over time. Here, a general approach to defining DMOP has been articulated. Therefore, a minimisation or maximisation problem has been formally defined by Equation 4.1:

$$\begin{aligned} \text{Min./Max. } F(x, t) &= \{f_1(x, t), f_2(x, t), \dots, f_M(x, t)\} \setminus x \in X^n \dots\dots\dots (4.1) \\ \text{s. t. } g(x, t) &> 0; h(x, t) = 0 \end{aligned}$$

Where,  $x$ ,  $f$ ,  $g$  and  $h$  represent decision variables, a set of objective functions that need to be minimised or maximised, inequality and equality constraints respectively.

In the context of decision space, the objective  $x^*(i, t)$  dominates over  $x(j, t)$ . In other words, the dominating one is better in at least one objective and not worst in any other objectives. This can be rewritten as in Equation 4.2:

$$F(j, t) < f(i, t)^* \setminus f(j, t) \in F^M \dots\dots\dots (4.2)$$

In this study, the dynamics of the environment is defined based on two parameters namely dynamic optimal Pareto front (DOPF) and dynamic optimal Pareto set (DOPS). DOPF can be represented with respect to the objective space as in Equation 4.3:

$$PF(t)^* = \{f(i, t)^* \mid \nexists f(j, t) \setminus f(i, t)^* < f(j, t) \in F^M\} \dots\dots\dots (4.3)$$

Dynamic optimal Pareto set with respect to the decision space at time  $t$  is denoted by  $PS(t)^*$  in Equation 4.4.

$$PS(t)^* = \{x_i^* \mid \nexists f(x_j, t) < f(x_i^*, t)^*, f(x_j, t) \in F^M\} \dots\dots\dots (4.4)$$

Moreover, the following  $\Delta f$  signifies how frequent the environment changes based on the actual PF, T(PF) and the obtained PF, O(PF) (Azzouz, Bechikh and Ben Said, 2017) as shown in the following Equation 4.5.

$$\Delta f = |T(PF) - O(PF)^*| \dots\dots\dots (4.5)$$

### 4.3 Defining the Dynamics of the DMOP

In order to solve the real-world DMOPs more effectively, it is necessary to take the following characteristics of the dynamics and problem types into account when designing new methodologies. There are several types to define dynamics in the literature for multi-objective optimisation. Frequency, severity and predictability are the prominent categories among them (Azzouz, Bechikh and Ben Said, 2017). Tantar *et al.* (Tantar, Tantar and Bouvry, 2011) proposed a classification model based on components that focus on the sources of the dynamism of the optimisation problem. They proposed the following four categories of dynamics.

- 1<sup>st</sup> order: dynamic parameter evolution modelled as  $H(F_\sigma, D, x, t) = F_\sigma(D(x, t))$
- 2<sup>nd</sup> order: dynamic function evolution modelled as  $H(F_\sigma, D, x, t) = D(F_\sigma, x, t)$
- 3<sup>rd</sup> order: dynamic state dependency evolution modelled as  $H(F_\sigma, T^{[t-j, t]}, x, t)$  given a transformation function T over time  $t$  and  $t - j$ .
- 4<sup>th</sup> order: Online dynamic evolution modelled as  $H(F_\sigma, D, x, t) = F_{D(\sigma, t)}(D(x, t))$

Where, H represents the model,  $F_\sigma$  is the multi-objective support function, D represents the vector of time-dependent functions where  $t$  and  $\sigma$  characterise time and the parameters. However, this model does not generalise the dynamic constrained problems which are associated with changing parameters, objectives and limitations.

On the other hand, (Farina, Deb and Amato, 2004) describes four types of Dynamic Multi-objective Optimisation Problems (DMOPs) according to the changes affecting the optimal PF and PS. In this study, the standardisation that is provided by (Farina, Deb and Amato, 2004) has been utilised which is mentioned in Table 4.1:

Table 4. 1: Dynamic MOP environment types

<b>PF(t)*</b>	<b>PS(t)*</b>	
	No Change	Change
No change	Type 4	Type 1
Change	Type 3	Type 2

The reason for using the above-mentioned standard is that it can deal with the changing PF and PS and this is required in both test cases that have been selected in this study. This standard can also provide the classification of the DMOPs and the sources of the dynamics which help to get the insights to analyse the performance of the proposed algorithm.

To define the optimisation functions in general in such a dynamic environment where the dynamics are controlled by a time-specific parameter  $\tau$  (i.e. a changing step when the problem changes) can be represented as in Equation 4.6:

$$\begin{cases} D_t(t(\tau), \tau) = t(\tau) + 1, & \text{when a change occurs in the environment} \\ D_t(t(\tau), \tau) = t(\tau), & \text{otherwise} \end{cases} \quad (4.6)$$

Therefore, the dynamic multi-objective optimisation problem in the finite period  $[1, \tau^{end}]$  can be defined as in Equation 4.7 (Raquel and Yao, 2013):

$$Optimise \left\{ \sum_{\tau=1}^{\tau^{end}} F_{\gamma(t_{\tau}, X_F^{G[1, \tau]})} (x_t) \right\} \dots \dots \dots (4.7)$$

Where,  $F$  represents a number of objective functions,  $X_F^{G[1, \tau]}$  is the set of solutions achieved by the applying algorithm  $G$  to solve  $F$  during  $[1, \tau]$ .

From the above discussions, the overall dynamics are set to design the proposed benchmark which leads to forming the first test case.

#### 4.4 Proposed Benchmark for a Dynamic Multi-objective Environment

In order to construct a dynamic environment in the RL settings, this study proposes the use of a grid-world which is a modified version of the traditional DST as described in Chapter 1 and 2. In this test case, the agent has to maximise its cumulative reward compared to the penalty that the agent will get while traversing the environment.

To extend this static DST environment and produce a dynamic DST, three different scenarios have been considered. The first environment is shown in Figure 4.1 which consists of different treasure values. These values change randomly over time. This randomness depends on the computers present time that helps to generate a random value in a finite horizon while the agent traverse in an episodic environment.

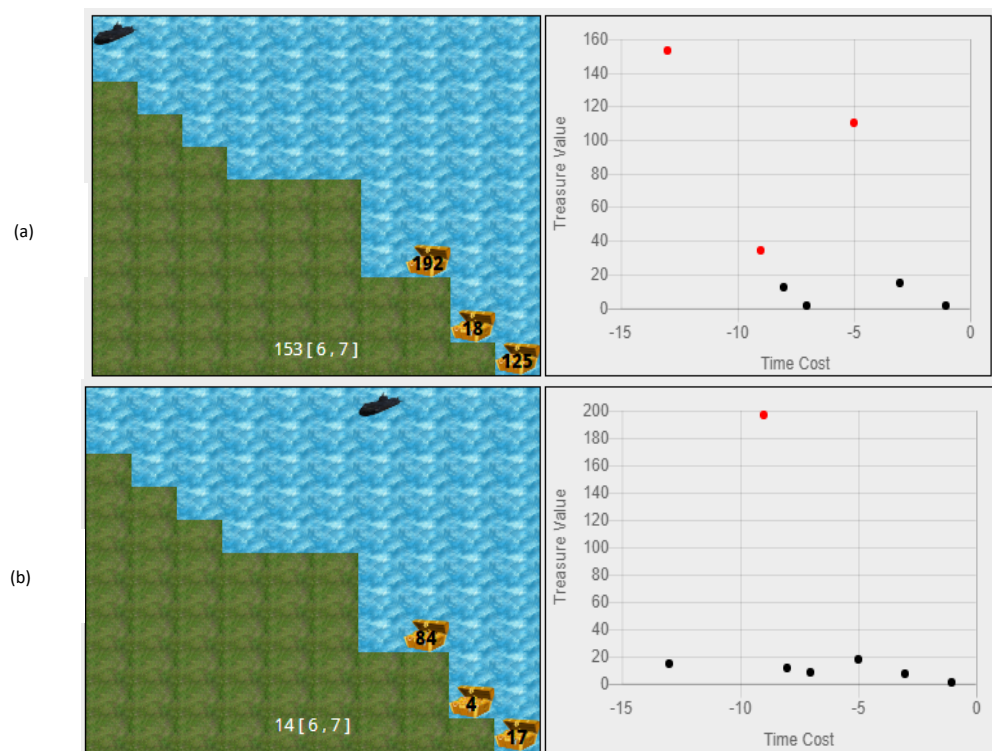


Figure 4. 1: Two instances (a, b) of dynamic deep-sea treasure (random treasure values-Type II)

Due to the random allocation of the treasure values, this environment can be categorised in type II where both  $PS^*$  and  $PF^*$  changes. This means that in every episode the true  $PF$  and  $PS$  will not be similar, and the agent cannot find any patterns that will help the agent to converge. Due to this randomness (i.e. ML incompatibility without having pattern) of the environment, this has been excluded for further implementation and analysis (Takeuchi, Kitahashi and Tanaka, 1975).

The next environment is similar to the random one. However, the treasure values follow a fixed amount of numbers which is categorised as silver (i.e. lower value) and gold (i.e. higher value). The agent dynamically collects the treasures from the grid world. This dynamic environment can be considered as Type III where  $PF^*$  changes and  $PS^*$  remains invariant. Figure 4.2 represents dynamic DST (silver and gold) environment where red and black dots represent identified gold and silver treasures respectively.

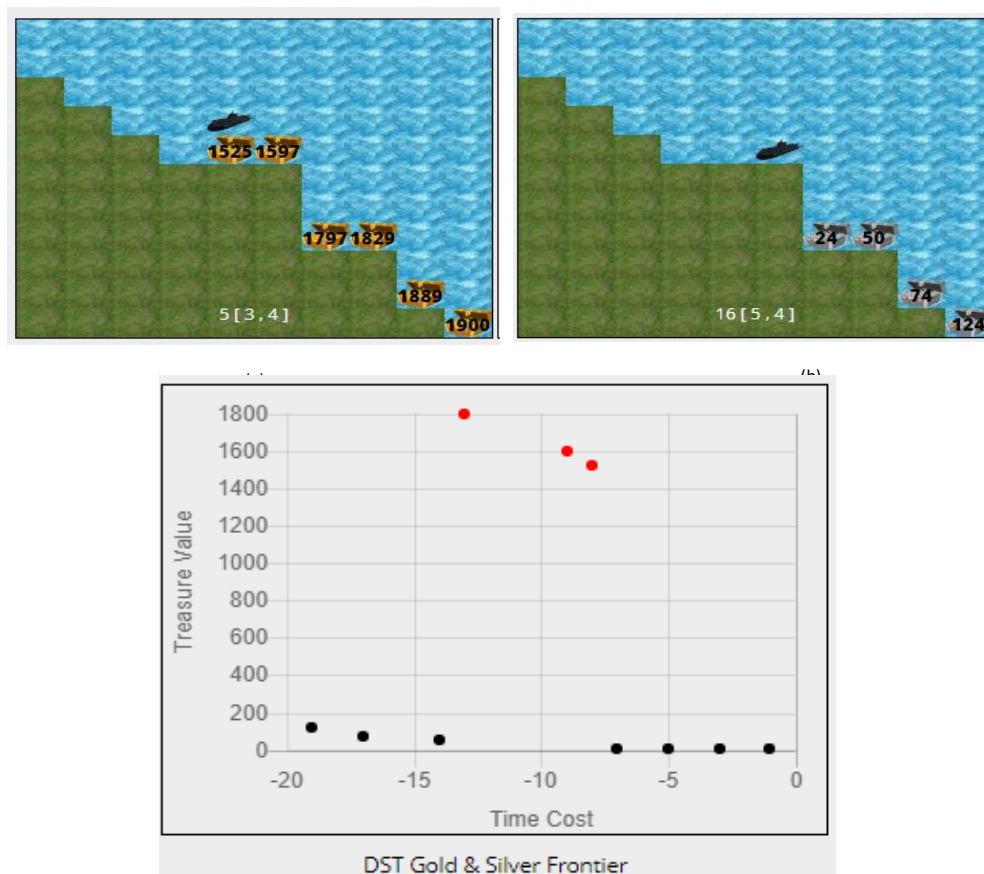


Figure 4. 2: Dynamic Deep-sea Treasure problem- Silver and Gold (Type III)  
(a) gold environment (b) silver environment (c) frontier

In the final environment, as shown in Figure 4.3, an enemy submarine (i.e. red submarine) has been introduced that attacks the agent (i.e. black submarine). The enemy submarine aims to hit the agent and in every clash between these two submarines results into the damage of the black agent's health meter by -2. This health parameter creates another objective that needs to be satisfied over time to survive.

This environment satisfies the dynamics of the environment in the category of type IV where both  $PF^*$  and  $PS^*$  remains unchanged. As the agent's vessel approaches, the enemy intends to hit the agent. At this moment, the priority of the black submarine is to save its life by not let the enemy damage its health meter and thus, a new objective is formed to be safe in addition to achieving the treasure. Hence, it has accommodated a different objective while traversing the environment.

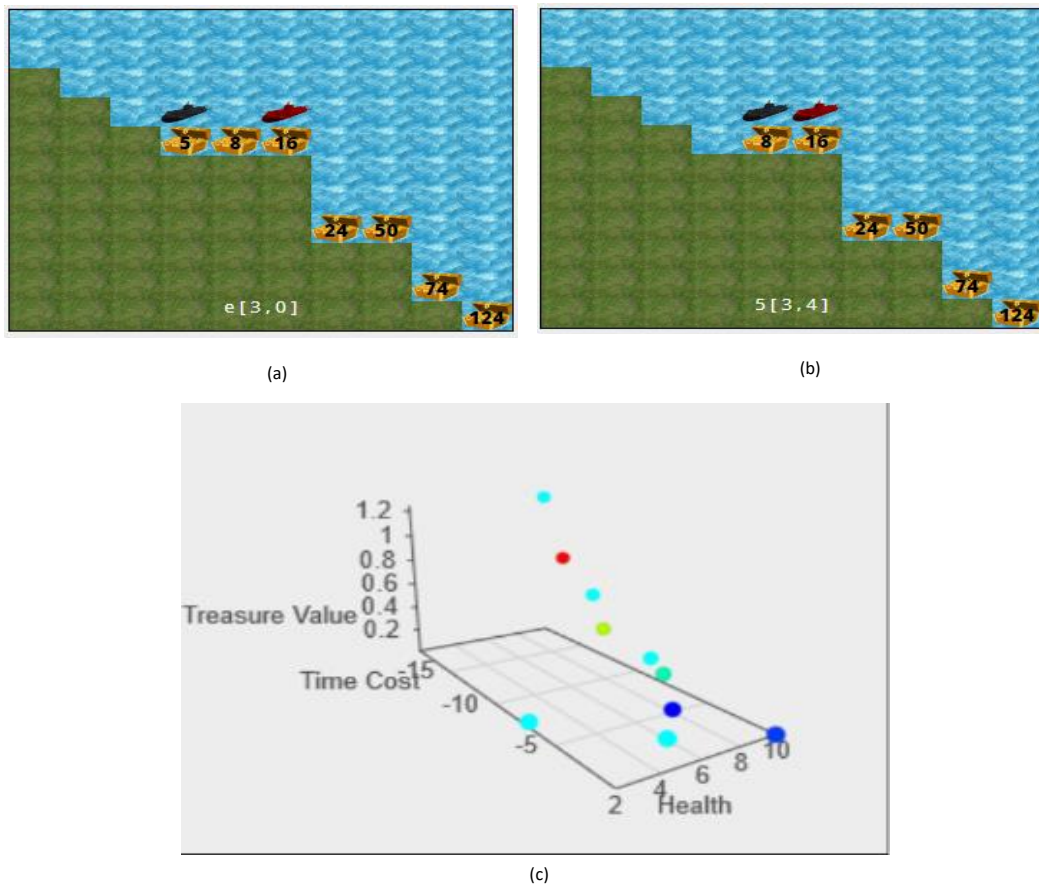


Figure 4. 3: Dynamic Deep-sea Treasure problem- DST Attack by Enemy (Type IV)



## 4.5 Empirical Setups for Test Case 1

### 4.5.1 The mathematical model for test case 1

At first, a conceptual model of this test case 1 has been created based on the mathematical model. To construct the model which is a 2D grid-world, the conventional approach to solve this sort of problems would be using the raw image as input for deep layer where the convolutional network and the output layer deals with images. The other approach is setting up the environment using raw code by formalising the MOMDP as argued in Chapter 3. Since another test case does not have any image input, both test cases are considered in a similar way. Therefore, the hardcode approach has been chosen by formalising the MOMDP. However, the latest success of DRL is based on the raw images by (Mnih et al., 2015) to play 49 Atari games with the same algorithm where the agent learns itself and achieves human-level expertise.

To get started with the concept of hard-code approach, let us consider a scenario as illustrated in Figure 4.4 which shows a sample of MDP. In this scenario, the agent needs to go from the root node to the destination node by satisfying all the objectives that are subject to the pre-defined constraints.

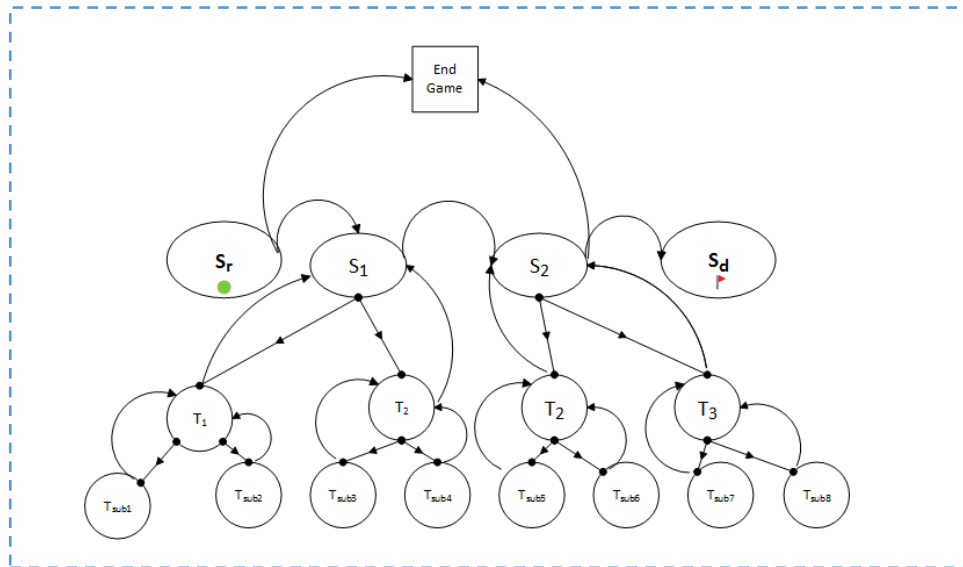


Figure 4. 4: A conceptual model of the Markov Decision Process (MDP) of the simulated environment

In the above Figure 4.4,  $S_r$  represents the source node where the traverse is started, and  $S_d$  is the destination node where the traverse will be ended. Each of the states is segmented with other objectives and criteria.

The following code snippets 4.1 and 4.2 show the reward allocations and the states respectively in the DST environment.

```
if scene is None:
    # While the scene is empty
    self._scene = np.zeros((11, 10))

    # Default Map as used in general MORL papers for DST
    self._scene[2:11, 0] = -100
    self._scene[3:11, 1] = -100
    self._scene[4:11, 2] = -100
    self._scene[5:11, 3:6] = -100
    self._scene[8:11, 6:8] = -100
    self._scene[10, 8] = -100
    # Rewards of the default map in the context of a static DST
    self._scene[1, 0] = 1
    self._scene[2, 1] = 2
    self._scene[3, 2] = 3
    self._scene[4, 3] = 5
    self._scene[4, 4] = 8
    self._scene[4, 5] = 16
    self._scene[7, 6] = 24
    self._scene[7, 7] = 50
    self._scene[9, 8] = 74
    self._scene[10, 9] = 124
```

Snippet 4.1: Rewards map in the DST environment

```
_state_map = {0: 0, 1: 1, 2: 2, 3: 3, 4: 4, 5: 5, 6: 6, 7: 7, 8: 8, 9: 9,
              10: 10, 11: 11, 12: 12, 13: 13, 14: 14, 15: 15, 16: 16,
17: 17, 18: 18, 19: 19,
              21: 20, 22: 21, 23: 22, 24: 23, 25: 24, 26: 25, 27: 26,
28: 27, 29: 28,
              32: 29, 33: 30, 34: 31, 35: 32, 36: 33, 37: 34, 38: 35,
39: 36,
              43: 37, 44: 38, 45: 39, 46: 40, 47: 41, 48: 42, 49: 43,
56: 44, 57: 45, 58: 46, 59: 47,
66: 48, 67: 49, 68: 50, 69: 51,
76: 52, 77: 53, 78: 54, 79: 55,
88: 56, 89: 57,
98: 58, 99: 59,
109: 60}
```

Snippet 4.2: States map in the DST environment

To illustrate the first test case mathematically, the components of the environment are given below:

- A submarine (i.e. black coloured) that works as an agent,
- the treasure values,
- time,
- rewards and
- health meter for the dynamic DST environment (attack by enemy)

Considering the objectives, the agent needs to hunt the highest treasures in the least possible time. However, for every step, the agent is penalised by -1 point. So, these two aims consist the conflicting objectives such as maximisation and minimisation for all the considered environments. In the third environment, an extra constraint has been introduced that is the health meter of the agent where the value of the meter is decreased by the hit of the enemy submarine. The followings are the list of the decision variables and constraints to formalise the objective functions.

Variable 1: Treasure values are divided into four categories as a tuple based on the defined benchmarks

1.  $T_{silver} = [1, 2, 3, 5, 8, 16, 24, 50, 74, 124]$
2.  $T_{gold} = [100, 925, 1231, 1442, 1525, 1597, 1797, 1829, 1889, 1900]$
3.  $T_{attack\ by\ enemy} = [1, 2, 3, 5, 8, 16, 24, 50, 74, 124]$
4.  $T_{random} = rand [1\ to\ \infty]$ , where only integer values are considered

In general, the treasure values can be expressed by the following Equation 4.8:

$$Treasure = (T_{i=0}^{i=\infty}, t)_{episode}; \text{ based on the environments ..... (4.8)}$$

Variable 2: For each episode  $E$ , the spent time by the agent can be expressed by Equation 4.9

$$Elapsed\ time = X_{cost(duration\ of\ traversing)}^E \dots \dots \dots (4.9)$$

*Variable 3: The health meter is decreased by  $j = -2$  from the pre – defined value  $n = 10$*

*by every hit by the enemy submarine as shown in Equation 4.10.*

$$health\ meter = \sum_{j=-2}^{n=j+(-2)} (H, t)_{episode} \geq 0 \dots\dots\dots (4.10)$$

*Variable 4: The reward function for the agent*

The reward function is defined by the vector  $\vec{R} = S \times A \times S'$  of  $n$  rewards in an MOMDP. This can be defined as following Equation 4.11 (Ruiz-Montiel, Mandow and Pérez-de-la-Cruz, 2017):

$$\vec{R} = \sum_k^{\infty} \gamma^k \vec{r}_{t+k+1} \dots\dots\dots (4.11)$$

Where, the agent gets a penalty of (-1) for every step during the traverse in the grid-world and  $\gamma \in [0,1]$  represents the discounted factor where the reward is received after  $k$  steps.

Hence, the two objective functions are defined as the following Equations 4.12 and 4.13:

**Objective 1**  $\rightarrow$

$$Minimise (f_1): \sum_{i=1}^n (X_{cost(duration\ of\ traversing)_i}^E, t); \forall_{episode} \dots\dots (4.12)$$

$$\mathbf{Objective\ 2} \rightarrow Maximise(f_2): (f_{treasure_{values}}, t); \forall_{episode} \dots\dots (4.13)$$

Subject to,

$$f(traverse) = \begin{pmatrix} 0 \times 0 & \dots & 0 \times 9 \\ \vdots & \ddots & \vdots \\ 10 \times 0 & \dots & 10 \times 9 \end{pmatrix}; \text{except the blocked areas}$$

The blocked areas in the grid-world are given below (see test case 1 in Chapter 1 for refreshing the orientation):

$$\begin{aligned} &(2 \times 0); \quad (3 \times 0 \dots 3 \times 3); (4 \times 0 \dots 4 \times 4); (5 \times 0 \dots 5 \times 5); \\ &(6 \times 0 \dots 6 \times 6); (7 \times 0 \dots 7 \times 5); (8 \times 0 \dots 8 \times 7); \\ &(9 \times 0 \dots 9 \times 7); (10 \times 0 \dots 10 \times 8) \end{aligned}$$

The time for changing the parameters are based on current time (pc clock) and the movement of the agent must not take place in the future (pc clock).

- (i) *iteration for converging  $\leq$  maximum limit of the epoch*

#### 4.5.2 Experimental settings for test Case 1

Considering the analysis of the algorithms, a set of hyperparameters are required to establish the experimental setup. This is also essential to make an impartial comparison for the considered algorithms. These hyperparameters need to be similar and the same for every case. To form the experimental setup for test case 1, the following network architecture has been followed as shown in Figure 4.5.

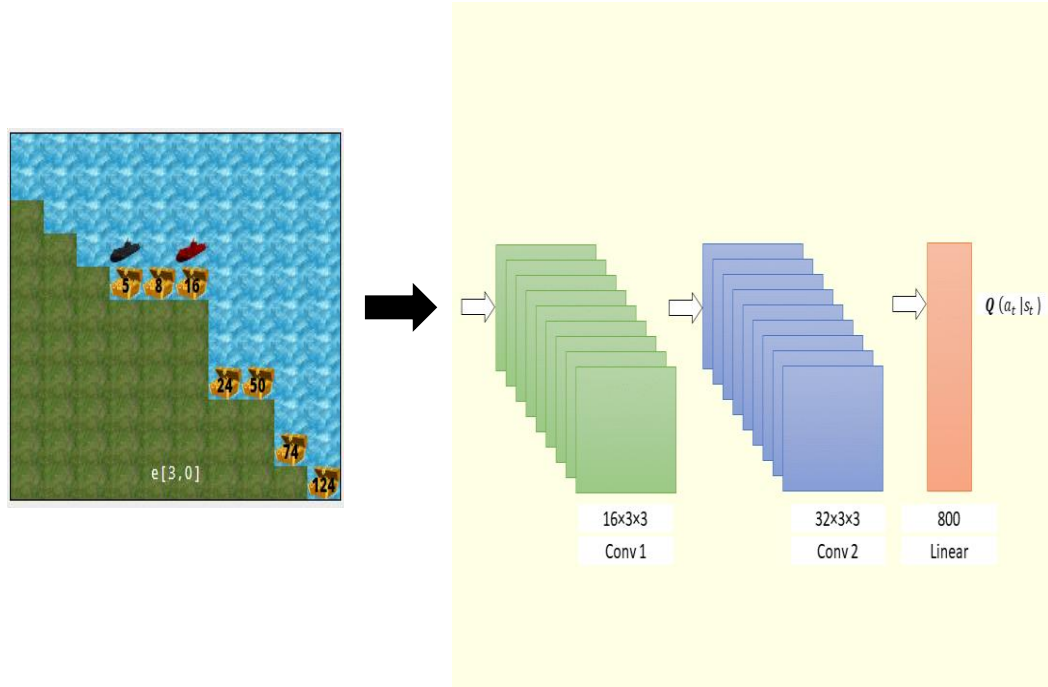


Figure 4. 5: Visualisation of the deep layer for the dynamic DST environments

In this structure as shown in Table 4.2, an experience replay, and a target network have been utilised to stabilise the network and the learning procedure. To reach all the states including the unvisited nodes, an  $\epsilon$ -greedy exploration policy has been implemented with annealing from 1 to 0.05. The discount factor is  $\gamma = 0.97$ , and the target network is reset in every 150 episodes. To stabilise the learning, mini-batches of 32 is used where the number of training chunk for each Adam update is computed. The replay memory size is set to 10K where Adam is sampled from recent actions. The optimisation algorithm of Adam of Keras implementation with a learning rate of 0.001 is used which is based on RMSProp (Wilson et al., 2017). For the experimental interval, the average steps have been counted after 1000 steps. Here, two convolutional layers of  $16 \times 3 \times 3$  and  $32 \times 3 \times 3$  with the rectified

linear unit (ReLU) are used as an activation function. Moreover, a fully connected layer has been established on the top of the two layers. See Appendix D for the whole setup including the decay, loss and the kernel functions architecture with the number of tensors. In addition, in the beginning, the number of “do nothing” actions have been set to 40 at the start of an episode.

Table 4. 2: Hyperparameters for test case 1

Parameter	Value
Learning steps	3 Million
Agent’s Evaluation (Interval)	1 Million
Replay memory size	10000
Target network update rate	1000
Learning rate ( $\alpha$ )	0.001
Exploration rate	0.4
$\varepsilon$ end step	1000
Discount factor	0.97
Optimiser	Adam
Batch size	32
No-op max	40

The following Figure 4.6 shows the visualisation of the graph edifice for the above-mentioned planning.

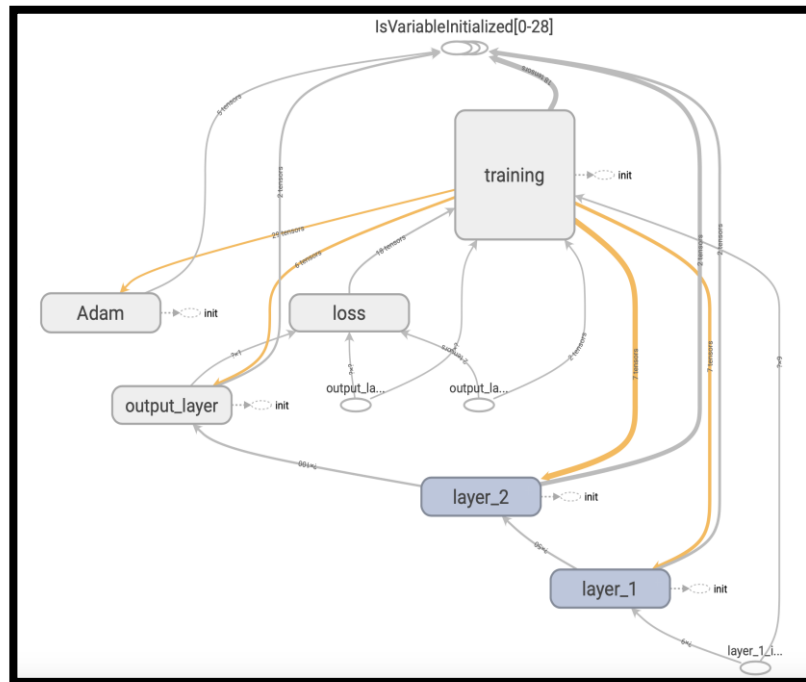


Figure 4. 6: Visualisation of the deep layer for test case 1 (extracted using TensorBoard)

## 4.6 Empirical Setups for Test Case 2

### 4.6.1 Mathematical model to formalise MOMDP for test Case 2

In the second test case, the whole environment has been formulated using MOMDP. During the conceptual design, a multi-criteria system has been defined which has different inputs such as  $I_1 = [IQA_1, IET_1, IVA_1]$  up to  $I_{22} = [IQA_{22}, IET_{22}, IVA_{22}]$ .

The first task was to form the MOMDP in this scenario based on the dataset which has been utilised using the multi-criteria selection mechanism. To do so, fuzzy AHP (FAHP) method has been used. The detailed description of the method can be found in (Hasan et al., 2016).

The external influences such as WQI values have been denoted by  $(\lambda_i; i=1:z)$ . To normalize, each set of attributes is defined by a global parameter of influence which is denoted by  $(\delta_1, \delta_2, \delta_3 \dots \delta_n)$  as mentioned in (Hasan et al., 2016; Klir and Yuan, 1995; Haiyunnisa, Alam and Salim, 2017). In this study, the criticality (of the zones) is set using the triangular fuzzy number (TFN) and scaled between 0 (resilient) and 1 (critical). In relation to the required fuzzy algorithm, set of membership functions have been used and defined by the triplet  $(l, m, n)$  as shown in Equation 4.14 and Figure 4.7.

$$U(x) = \begin{cases} (x - l) / (m - l), & l \leq x \leq m \\ (n - x) / (n - m), & m \leq x \leq n \\ 0, & \text{otherwise} \end{cases} \dots \dots \dots (4.14)$$

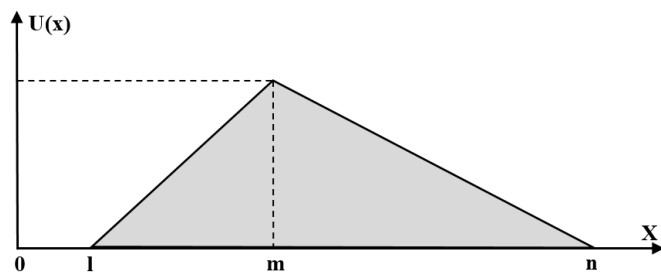


Figure 4. 7: Distribution of triangular fuzzy number (Wang, Wang and Qi, 2016)

where,  $l$  denotes the smallest likely value;  $m$  denotes the most probable value; and  $n$  denotes the largest possible value of any fuzzy event. In other words,  $m$  is the

most possible value of the fuzzy number  $U(x)$ ,  $l$  and  $n$  are the lower and upper bounds, respectively. Table 4.3 outlines the results of the pair-wise comparisons matrix. In order to show the importance intensity of each criterion, factors ( $\lambda$ ) and sub-factors ( $\sigma$ ) are utilised using TFN.

Table 4. 3: Global index of importance intensity of WQI by AHP and TFN (Bahri, Ben Amor and El-Ghazali, 2014)

IQA/ IVA			IET		
Linguistic variable to select the decision	Crisp scale of AHP	Scale of TFN	Significance intensity for IET	Scale in Traditional AHP	Scale of TFN
Very poor quality ( $\lambda_1$ )	1	(0.1,0.2,0.3)	High oligotrophic/High quality( $\sigma_1$ )	6	(0.9, 1.0,1.0)
Poor quality ( $\lambda_2$ )	2	(0.3,0.4,0.5)	Oligotrophic/Good quality( $\sigma_2$ )	5	(0.8,0.9,1.0)
Average quality ( $\lambda_3$ )	3	(0.5,0.6,0.7)	Mesotrophic/Average quality( $\sigma_3$ )	4	(0.7,0.8,0.9)
Good quality( $\lambda_4$ )	4	(0.7,0.8,0.9)	Eutrophic/Average Quality( $\sigma_4$ )	3	(0.4,0.5,0.6)
High quality( $\lambda_5$ )	5	(0.9,1.0,1.0)	Highly/Poor quality eutrophic( $\sigma_5$ )	2	(0.0,0.1,0.2)
			Very highly eutrophic/Very poor quality( $\sigma_6$ )	1	(0.0,0.1,0.2)

Equation 4.15 calculates the global significance of factor and sub-factor for selecting the critical zones based on the parameters (i.e. IQA, IET and IVA) and their threshold values which have been discussed in Chapter 3:

$$\delta = \sum_{i=1}^z (\lambda_i \times \sigma_i) \dots \dots \dots (4.15)$$

where,  $\delta$  denotes global significance;  $\lambda$  and  $\sigma$  are the significance of factors and sub-factors, respectively; and  $z$  denotes the number of zones. Therefore, each WQ monitoring station can be expressed by Equation 4.16:

$$E_{i=1}^k = \{(IQA_z, \sum_{i=1}^z \delta_i^k), (IVA_z, \sum_{i=1}^z \delta_i^k), (IET_z, \sum_{i=1}^z \delta_i^k)\} \dots \dots \dots (4.16)$$

Where,  $S_{i=1}^k$  denotes a set of alternatives,  $Z = [E_1^k, E_2^k, \dots \dots \dots, E_n^k]$

Hence the two objective functions are defined in Equation 4.17 and 4.18:



**Objective 1** →

$$\text{Minimise } (f_1): \sum_{i=1}^n ((IVA_z, \sum_{i=1}^z \delta_i^k, t) \& (IET_z, \sum_{i=1}^z \delta_i^k), t); \forall_{episode}. \quad (4.17)$$

$$\text{Objective 2} \rightarrow \text{Maximise } (f_2): ((IQA_z, \sum_{i=1}^z \delta_i^k), t); \forall_{episode} \dots \dots \dots (4.18)$$

Subject to,

$$i) f(\text{traverse}) = \begin{pmatrix} 0 \times 0 & \dots & 0 \times 21 \\ \vdots & \ddots & \vdots \\ 21 \times 0 & \dots & 21 \times 21 \end{pmatrix}; \text{ including all the associated stations}$$

The time for changing the parameters are based on current time (pc clock) and the movement of the agent must not take place in the future (pc clock).

$$ii) \text{ iteration} \leq \text{maximum limit of the epoch}$$

The total weight of each criterion is determined using fuzzy mean, as shown in Equation 4.19:

$$\bar{w} = \left[ \frac{l_i}{\sum_{i=1}^z n_i}, \frac{m_i}{\sum_{i=1}^z m_i}, \frac{n_i}{\sum_{i=1}^z l_i} \right] \dots \dots \dots (4.19)$$

where,  $\bar{w}$  denotes global fuzzy mean;  $l, m$  and  $n$  are the membership functions to select a particular criterion (i.e. critical zone or station); and  $z$  represents the number of zones.

In this step, the total weight of each criterion (i.e. WQI's resilience) should be defuzzified before normalisation of the results of global fuzzy mean in crisp values (which are subject to fuzzy sets and the corresponding membership degrees). This process interprets membership degrees into a specific decision (Rondeau et al., 1997). Equation 4.20 is used to defuzzify the total weights. The total weights are used to rank the stations in different zones.

$$W_{id} = \left[ \rho \left( \frac{\alpha(w_{in})}{\sum_{i=1}^z \alpha_i} \right) \times (1 - \rho) \left( \frac{\alpha(w_{il})}{\sum_{i=1}^z \alpha_i} \right) \right]; \rho \in [0,1] \dots \dots (4.20)$$

where,  $W_{id}$  denotes the total weight of each criterion,  $\rho$  represents optimism index (i.e. reflects risk-taking attitude of decision makers');  $\alpha$  denotes the cut values in

the crisp;  $w_{in}$  and  $w_{il}$  are the upper and lower bounds of the  $\alpha$ -cut values, respectively (Klir and Yuan, 1995; Kim and Park, 1990) determined using the weighting average method. It should be noted that the factors belonging to the IQA and IVA have five  $\alpha$ -cut values and IET have six cut values.

Finally, in this step critical zones are determined using Equation 4.21 and Equation 4.22. The higher weight values represent more critical and the lower ones represent the less critical zone.

$$f_{v_{WQI}} = \sum_{i,p=1}^z (w_{id} \times \rho_p) \dots\dots\dots (4.21)$$

$$Z = \sum_{i=1}^z f_{v_{WQI_i}} \dots\dots\dots (4.22)$$

where,  $f_{v_{WQI}}$  represents variable alternatives (i.e. critical stations in each zone) for each WQI individually; WQI could be either IQA, IET or IVA;  $W_{id}$  denotes the total weight of each criterion;  $\rho$  denotes optimism index;  $p$  represents the number of iteration of the satisfaction level; and  $z$  represents the total number of critical stations;  $Z$  denotes the zone.

Now, the procedure of formalising the multi-objective environment in the RL settings for this test case has been explained. For a comprehensive evaluation of RL, readers are referred to (Sutton and Barto, 2018). To solve the RL in this context, Markov Decision Process (MDP) has been defined as the collection of the following components (see terminology and its explanation for an MDP):

- States:  $S$
- Actions:  $A(s), A$
- Transition model:  $T(s, a, s') \sim P(s'|s, a)$
- Rewards:  $R(s), R(s, a), R(s, a, s')$
- Policy:  $\pi(s) \rightarrow \alpha\pi^*$  is the optimal policy

In this case of an MDP, the environment is partially observable because of the dynamics of the environment. Thus, the agent needs a memory (i.e. experience replay) to store the past observations to make the best possible decisions. The

problem settings will be clearer to understand by analysing the Markov property as below.

In the Markov property, the information of the near future (i.e. at time  $t + 1$ ) depends on the present information at time  $t$ . A sequence of the zones  $[z_1, z_2, \dots, z_{22}]$  in the selected state follows the first order of Markov property as expressed by Equation 4.23 (Silver, 2015):

$$P(z_t | z_{t-1}, z_{t-2}, \dots, z_1) = P(z_t | z_{t-1}) \dots \dots \dots (4.23)$$

where  $z_t$  depends only on  $z_{t-1}$ . Therefore,  $z_{t+1}$  will depend only on  $z_t$ .

However, in this scenario, the Equation can be expressed by Equation 4.24.

$$P(z_t | z_{t-1}, z_{t-2}, \dots, z_1) = P(z_t | z_{t-1}, z_{t-2}) \dots \dots \dots (4.24)$$

where,  $z_t$  depends on  $z_{t-1}$  and  $z_{t-2}$ .

Thus, the stations can be converted to a Markov property if the probability of the new state depends on the next state and claim that  $z_{t+1}$ , depends on the current state,  $z_t$ , such that the current state captures and remembers the property and knowledge from the past. Therefore, as per the Markov property, the grid-world (i.e. the environment) is considered to be stationary (Feinberg and Shwartz, 2014). However, in this scenario, type III dynamics exist in this environment as mentioned in (Farina, Deb and Amato, 2004).

A finite block of 22 zones is considered with their corresponding data stations that are changing based on the monthly data. For each station, the velocity of the data has been formulated in a continuous manner where the data changes at next time step  $t + 1$  that is only determined by the current system state. It is independent from the previous states due to the changing values of the monthly data such as weather (e.g. drought, precipitation) and man-made changes such as usages and contamination as mentioned in Chapter 2 and 3. Therefore, the stations of each zone can be treated as an MOMDP (i.e. see a sample in Figure 4.8).

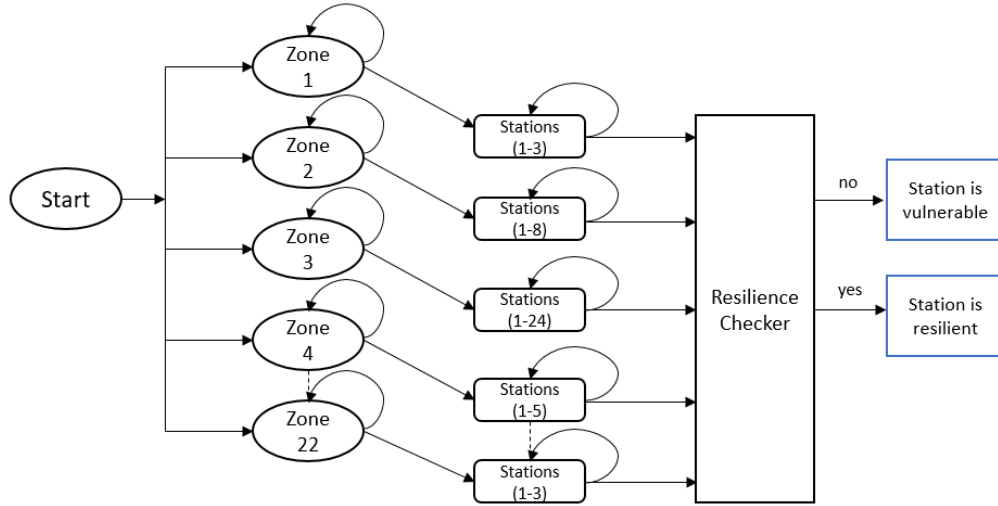


Figure 4. 8: MDP for the resilient area selection

Here, the key concepts to facilitate the DRL-based WQR prediction have been formulated. The terminal of each zone is related to the dataset, which is chosen from the discrete levels, denoted as  $Z = \{Z^1, Z^2, \dots, Z^n\}$ . Therefore, the entire action space is denoted as:  $A = \{A^1, A^2, \dots, A^n\}$ . On the other hand, the action space is determined based on the observation of the multi-objective optimisation and the selection on the actions based on the satisfaction of the objectives (maximising IQA and minimising IET, IVA). In this work, the current (physical) time and the values of IQA (i.e. the higher the better) and IET and IVA (i.e. the lower the better) are considered to determine the optimal control action.

Incorporating the current time information in the state enables the DRL algorithm to adapt the time-related activities, such as time-varying data for months that changes across the year. This is important because the weather pattern varies significantly throughout the year and thus, the recorded dataset reflects the changes on a monthly basis. Hence, it is required to facilitate the system and give the flexibility for the MOMDP to adapt to time-variant variables.

Considering a reward function, the agent does the job by taking a sequence of actions  $a_{t-1}$  at state  $s_{t-1}$ , the block will evolve into a new state  $S_t$  and the DRL algorithm will receive an immediate reward  $r_t$  in the MOMDP, the reward function  $\vec{R} = S \times A \times S'$  is a vector of n rewards rather than a scalar with an element for

each objective. Similarly, the reward function is also the vector value such as  $\vec{R} = (s, a, s')$ . This can be defined by Equation 4.11 as mentioned in the first test case. The rewarding factor includes a penalty of (-1) for detecting the wrong station which is not resilient and (+1) for detecting the right one.

During the traverse in the block, the target is to maximise the accumulative reward  $R$ , where  $\gamma = [0,1]$  is a decay factor. Agent's target is to find the optimal policy ( $\pi^*$ ), which maximises the projected reward. The optimal policy, in this case, is responsible to maximise the amount of reward received or expected to receive over a lifetime. Thus, in this method, the policy is nothing but a guide to direct which action needs to be taken in a given state. The agent determines the resilient area with the highest IQA values compared to the lowest IET and IVA as mentioned in the threshold values in Chapter 3 and Appendix B.

The optimal value  $Q^*(s_t, a_t)$  is used to represent the maximum accumulative vector reward to determine all the Pareto solutions that can be calculated by the Bellman Equation (Gross, 2016) as shown in Equation 4.25.

$$\vec{Q}_n^*\{(s, a), t\} = \begin{cases} (1 - \alpha_n)\vec{Q}_{n-1}(s, a) + \alpha_n[\vec{r}_n + \gamma V_{n-1}(s'), t]; & \text{if } s = s_n \wedge a = a_n \\ \vec{Q}_{n-1}\{(s, a), t\}; & \text{otherwise} \end{cases} \dots (4.25)$$

$$\text{Where, } V_{n-1}(s) = \max_{a \in A} \vec{Q}_{n-1}\{(s, a), t\}$$

In this test case, the value has been estimated by a Q-learning method as described in (van Hasselt, Guez and Silver, 2015). A high-level structural design of the RL agent that interacts within the environment is given below as shown in Figure 4.9.

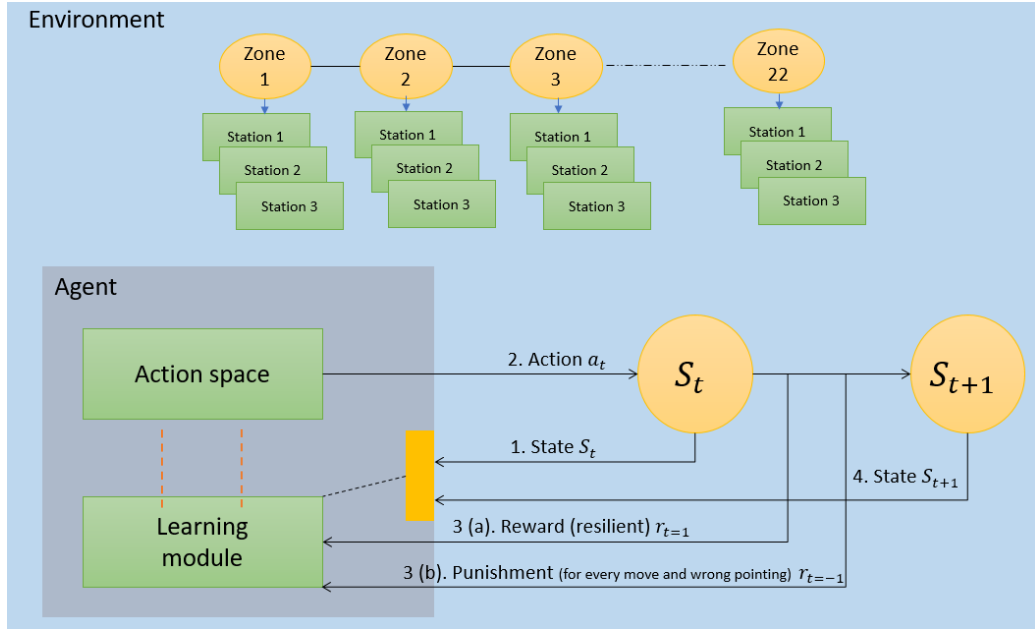


Figure 4. 9: High-level architecture of the RL implementation on the WQR dataset

To dig into deeper, let us consider the following environment and the given information. The agent in the MOMDP needs to identify the resilient zones based on the IQA, IET and IVA. The following visualisation (presented in Figure 4.10) shows a conceptual understanding of how the agent traverses in the environment and lists the resilient areas. The MOMDP represents a grid-world as shown in the visualisation. The filled circle shows the critical zones whereas the white circles show the resilient areas.

This grid-world is formed into states (i.e. zones with stations), actions (i.e. which zone to traverse), transition models (i.e. the selection from one zone to another) and rewards (i.e. achieved by identifying the resilient zone). The solution for this MOMDP is an optimal policy that relies on the vector rewards which determines the critical as well as the resilient zones. These rewards are connected to the objectives to find the optimal policy. Since, the dynamic nature and partially observable MOMDP, the agent requires an experience replay as shown in the Figure 4.11 to store the past observations to make the best possible decisions which are close to the Pareto Front.

Consequently, the continuous input from the agent can play an important role in state formation. Hence, the state spaces can be either discrete or continuous. The agent starts from the start state  $S_1$  and must reach the resilient zone by calculating the parameters (e.g. IQA, IET and IVA).

Consider the following grid-world as having 22 discrete states, where the orange-colour grid is the goal state, and the black one is the agent that needs to identify the resilient zone and distinguish them in a stack. The states are represented by the coordinates of  $(x, y)$  whereas the actions are the execution by the agent in a particular state. In other words, actions are sets of things that an agent is allowed to do in the given environment.

Consider this example, 22 discrete states with 4 discrete actions such as up, down, right and left. Therefore, the action,  $a \in A$  where,  $A = \{up, down, right \text{ and } left\}$ . Needless to mention, this action can be treated as a function of the state, that is,  $a = A(s)$ , where depending on the state function, it decides the best possible action in the current state based on the vector rewards. The transition model is defined by the current state  $(s)$ , action  $(a)$ , and the new state  $(s')$ . This can be represented as  $T(s, a, s')$  that defines the rules to traverse in the environment. It gives the probability  $P(s'|s, a)$  for the new state  $s'$ .

Let us consider the following information is pre-defined for this environment:

- Agent traverse between the Zones  $Z_1$  to  $Z_{22}$  and their corresponding stations within a grid-world.
- Once the agent reaches the goal state  $G$  (critical stations), it receives a reward of +1. However, for every step, it gets -0.04 rewards.
- The agent gets -1 for selecting the wrong station  $C$ .
- Thus, these two states (i.e.  $G$  and  $C$ ) are the terminal states, by reaching any of these two, the traverse is over. If the agent encounters the  $G$  state,

the agent wins, while if it enters the other one, then the agent loses the game.

- Discounted factor  $\gamma = 0.9$ , the utility at the first-time step is 0, except for the G and C states.
- Transition probability  $T(s, a, s')$  is equal to 0.8 if the agent traverses the preferred direction; otherwise, 0.1. For example, if the optimal location is up and the agent does the same then the action's probability is 0.8 otherwise for every movement the probability is 0.1.

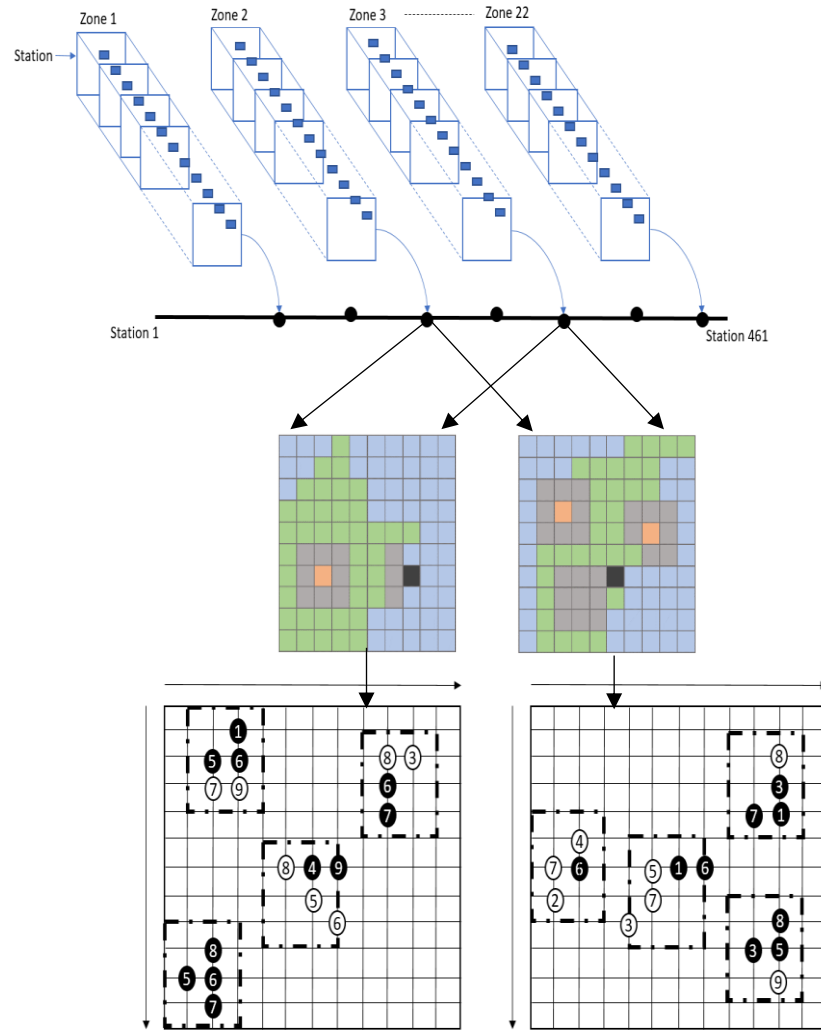


Figure 4. 10: A visualisation of knowledge gathering by the agent based on the resilient areas



#### 4.6.2 Experimental settings for test case 2

Similar to the first test case, an identical setting of the hyperparameters has been used for a fair comparison and the experimental setup for all of the considered algorithms. The following Table 4.4 shows the list of hyperparameters that has been used for second test case to determine the vulnerable zones based on the water quality resilience in the RL settings.

Table 4. 4: Hyperparameters for test case 2

Parameter	Value
Learning steps	3 Million
Agent's Evaluation (Interval)	1 Million
Replay memory size	10000
Target network update rate	1000
Learning rate ( $\alpha$ )	0.001
Exploration rate	0.4
$\epsilon$ end step	1000
Discount factor	0.9
Optimiser	Stochastic Gradient Descent (SGD)
Batch size	32
No-op max	40

Where, the network architecture has 4 convolutional layers. The first convolutional layer is devised by 32 filters of size 8x8 and a stride of 4. Also, the first layer is followed by 2 layers with 64 filters of size 4x4 and a stride of 2. The last convolutional layer comprised of 64 filters of size 3x3 and a stride of 1. These convolutional layers are followed by a 512 units of a fully connected layer.

Furthermore, the fully connected layer is responsible for projecting to the output of the network (i.e. the Q-values). All these layers are detached by Rectifier Linear Unit (ReLU) activation function. The reason for using ReLU is the sparsity and fewer chances to vanish the gradient. The only exception is the output layer which has linear activation.

In this deep layer, stochastic gradient descent (SGD) optimisation technique is used. Finally, the output layer is connected to multiple groups of nodes. The number of groups is identical to the number of objectives (e.g., time vs. treasure value or resilient data). Each group comprises a number of nodes that corresponds to the number of possible actions which leads to governing the policy. Technically, the DQN architecture has been created using TensorFlow and Keras. The visualisation and fine-tuning have been done using TensorBoard.

Given the RL framework using MOMDP discussed above, the optimum Q value is needed to find out from different deep Q networks for all actions  $a \in A$ . These DQNs represent the state-actions portfolio that needs to be selected by an optimal policy as described in (Silver et al., 2016). A convolutional neural net has been used as the Q network estimator to separate state-value and actions. State-action networks learn the Q-values for each action  $a_1$  given a state  $S_1$ , at some particular time step  $t$  by minimising the temporal difference error.

With this structure, the Q-value estimates for all control actions for each objective in different states. Performing one forward pass (inference) in the neural network improves efficiency when selecting actions with the  $\varepsilon$  –greedy policy. The linear layer is used for conjecturing action value at the output.

Figure 4.11 shows the network architecture with episodes  $\{e_1, e_2, \dots, e_r\}$  and loss function integrating into the whole process to update the Q network for each episode from the emulator. This also shows the utilisation of the experience replay for vector reward for each action in the MOMDP.

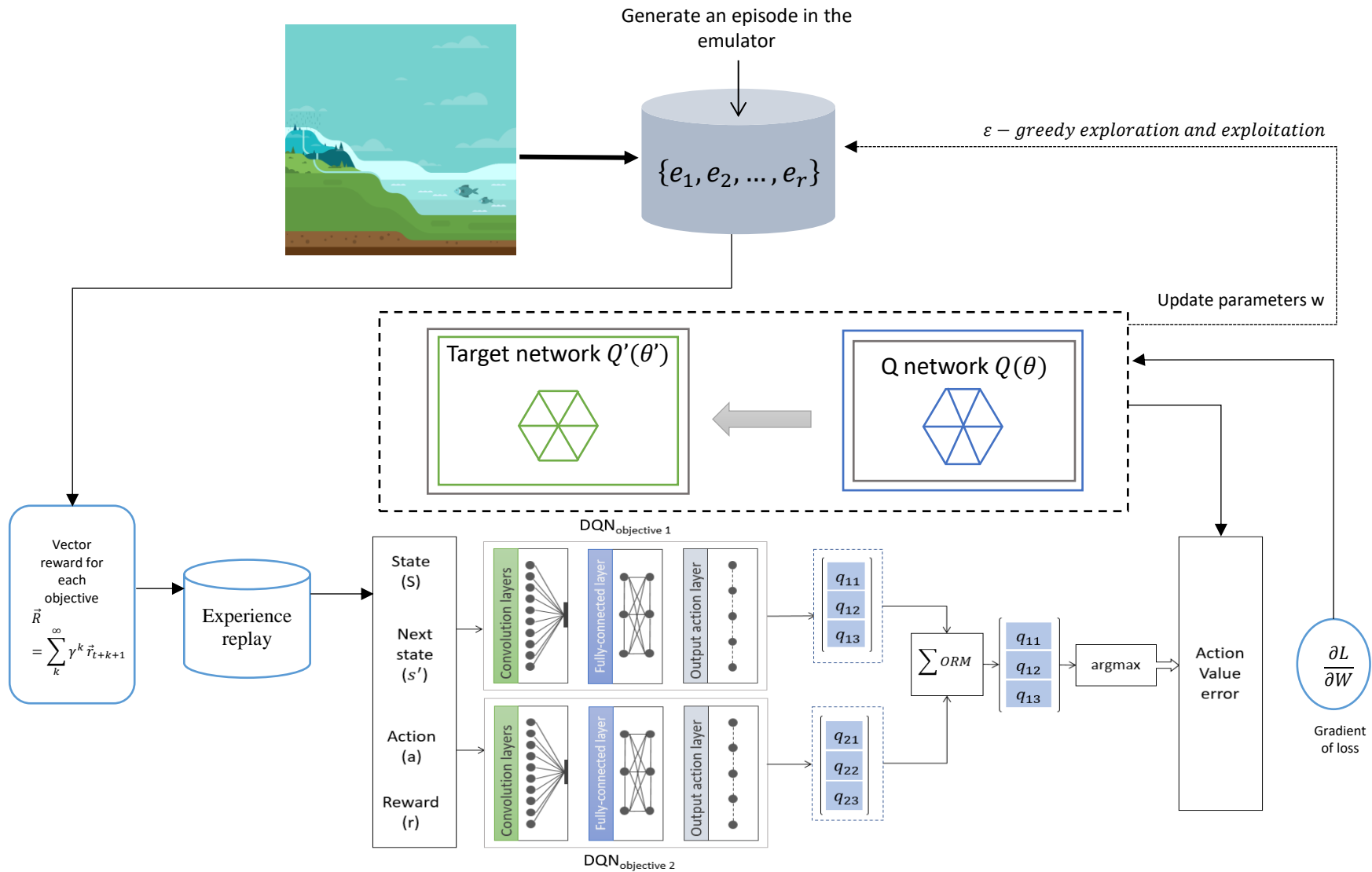


Figure 4. 11: Structure of the DRL framework that is used in test case 2

The gradient of loss as an error function has utilised between the target Q value and the actual one. The underlying principle is that while the agent has found a resilient zone based on IQA, IET and IVA, it does not reveal the worse control actions. Thus, the agent focuses on finding better control actions. As shown in Figure 4.8, the one-step state transition process is represented by a tuple of previous state ( $S_{t-1}$ ), previous action ( $a_{t-1}$ ), immediate reward  $r_t$  and current state  $S_t$ . The target vector of the neural network is calculated by Equation 4.26.

$$\pi^*(a | s_1) = \begin{cases} target - val(s_{t-1}, a) \geq 1; & \text{if } a = \operatorname{argmax} Q^*(s_t, a), \text{ where } \forall_{actions} \\ 0; & \text{otherwise} \end{cases}$$

$$V^{\pi^*}(s_{t+1}) = \max_{a \in A_{t+1}} Q^{\pi^*}(s_{t+1}, a) \dots \dots \dots (4.26)$$

Figure 4.12 shows the overall graph structure based on the TensorFlow that is produced by the TensorBoard for the above-mentioned network.

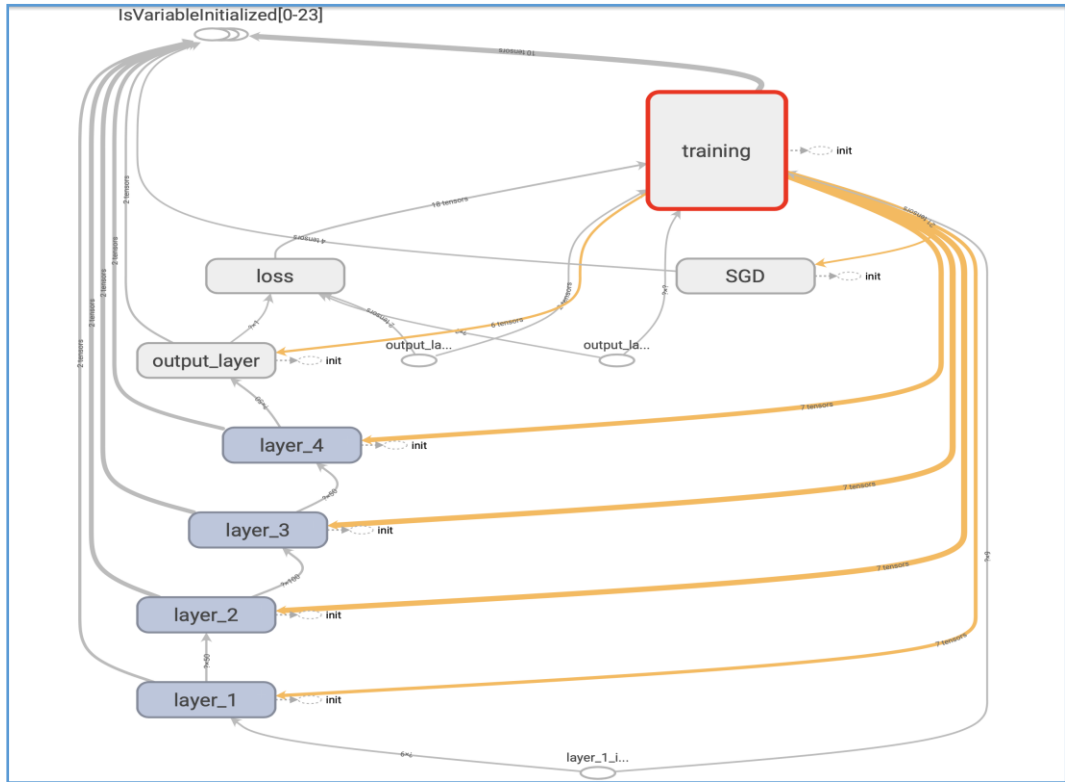


Figure 4. 12: Visualisation of the deep layer for test case 2 (extracted using TensorBoard)

For other control actions, the target value is set to the estimated current value of that action. Each DQN is responsible for approximating the Q-value for a particular policy (Tajmager, 2017). At each time step, all networks will receive the state information and then determine the control action separately. A brief overview of policy selection mechanism is shown in Figure 4.13. However, a detailed discussion has been mentioned in Chapter 5.

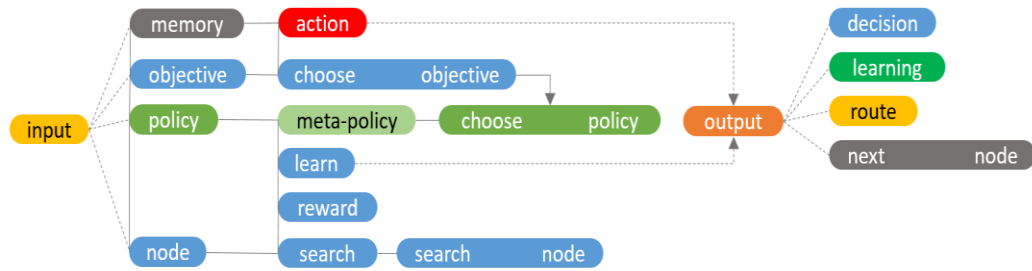


Figure 4. 13: Parity Q deep Q network-based Policy selection

## 4.7 Summary

Problem settings are always crucial in the study of the DMOP. Moreover, developing a dynamic environment in the context of RL is also a challenging task. In this chapter, an overview of the problem settings and their definition as well as experimental setups are presented. Here, the essential components for designing the benchmark and how it has been defined as a dynamic benchmark is described. Besides, problem definition and mathematical models of the considered test cases are explained.

In a nutshell, this chapter delivers 4 environments which are dynamic in nature. They are as follows:

1. Dynamic DST (silver and gold),
2. Dynamic DST (random),
3. Dynamic DST (attack by enemy) and
4. Dynamic WQR environment.

In summary, the followings can be drawn regarding the DMOP in this chapter:

- a. Optimisation goal in the DMOP: In the context of the RL settings, the proposed benchmark satisfies the primary goal of the benchmark which has got the several challenges in terms of changing parameters including the conflicting objectives.
- b. Constraints: Overall, the constraints of the benchmark have been identified clearly which have helped to form the mathematical model.
- c. Detectability of the changing optima: In every scenario of the proposed benchmark, the changing optima needs to be detected by the enforced algorithm. Therefore, this benchmark satisfies the dynamic behaviour in the multi-objective environment.

Finally, as discussed earlier in Chapter 2, many DMOPs studies are not related to the real-world problem. In this study, the real-world problem has been addressed and formulated the MOMDP in the context of the changing parameter dynamically in the RL settings. These changing parameters of test case 2 are based on the changing PF and PS. This chapter also highlights to visualise the network architecture. In the context of the real-world situation, the problem settings can be related to some other real-world and complex scenarios as mentioned in (Nguyen, 2011).

## Chapter 5

### Proposed Algorithm: Parity Q Deep Q Network (PQDQN)

#### 5.1 Introduction

In this chapter, a novel approach for computing the Pareto frontier in the MOMDP using MORL has been discussed. In this method, typical features of the RL method are integrated such as state, action, and reward (i.e. vector reward for multi-objective) based on MOMDP as discussed in the previous chapter. The working procedure of the algorithm is also deliberated in this chapter.

Here, this method makes the PF more beneficial as a decision support system by mapping different objectives and balance them. The method also allows to select the ideal multi-objective preference given by the assured boundaries with self-learning and mapping among different objectives. It is worth mentioning that, in this method, the boundary of the problem has set in the finite horizon. The problem also considers as ergodic and controllable MDP. The vector rewards procedure that has been applied is similar to the method that is described in (Roijers et al., 2013). For the optimisation technique, the Adam and the stochastic gradient approach (SGD) have been implemented in test case 1 and test case 2 respectively. The optimisation processes include several targets such as identifying the changing optimal and the distance from the obtained PF and the true PF. The decision-making process explored in this proposed method corresponds to the object-relation mapping that is a combination of Q values provided by the different DQNs.

In general, the aim of this chapter is to develop an algorithm which can ensure convergence and diversity for the defined problems that need to be addressed in the context of DRL settings. To achieve this target, a well-established reinforcement learning can be approached such as dynamic programming, Monte Carlo tree search (MCTS) and temporal difference learning (TD). To solve the

defined problem, the solution needs to find all the non-dominated policies at once as explained in (Barrett and Narayanan, 2008). Considering the simultaneous multi-policy search and updating the current state, a temporal difference learning mechanism has been chosen that is a model-free and off-policy algorithm such as Q learning. Moreover, it has been considered because of its capability to determine the optimal action-value function which learns from the reward function in an MDP that is compatible with the considered two test cases.

## **5.2 Deep Q Network (DQN) Selection**

Both DL and DRL are the ML techniques that learn autonomously. However, the difference between DL and DRL lies in the learning procedure followed by the agent. While DL requires a training dataset and then applying that learning to a new dataset, DRL learns by interacting within the environment by adjusting actions based on the feedback to maximise the highest expected reward. However, these two techniques are not mutually exclusive. To implement the DRL mechanics using deep Q networks (DQN), some strategies need to be followed such as selecting the double DQN or dueling DQN, asynchronous DQN and so on (Thomas Simonini, 2018).

Mnih et al. (2015) introduced Deep Q-Network and ignited the area of RL. The deep Q networks have been critically reviewed and the drawbacks of using Deep Q networks such as incompatibility in the non-Markov model (i.e. next state always relies on the current state), overestimation for continuous action space, overfitting and poor exploration due to sparse feedback (Mnih et al., 2013) are considered. These characteristics of DQNs are always important before applying to solve a problem. The common extensions of the DQN architecture are double DQN (Van Hasselt et al., 2016), multi-policy DQN (Nguyen, 2018), multi-objective Monte Carlo Tree Search (Wang et al., 2012), multi-policy Q learning (Ruiz-Montiel, Mandow and Pérez-de-la-Cruz, 2017), prioritised experience replay (Schaul et al., 2015), duelling architecture (Wang et al., 2016). In addition, (Anschel, Baram and Shimkin, 2017) proposed an average of previous Q-values



estimation to reduce inconsistency and instability. He et al., (2016) proposed a way to accelerate DQN by optimality tightening to propagate reward more rapidly as well as improve accuracy over DQN.

The following algorithms shown in Table 5.1 are considered in this study based on the policy evaluation in DQNs:

Table 5. 1: Comparison of the analysed algorithms

Item	Algorithms				
	Deep Q Network (DQN)	Double DQN (DDQN)	Multi-policy DQN (MPDQN)	Multi-objective Monte Carlo Tree Search (MO-MCTS)	Multi-Pareto Q Learning (MPQ)
Support Multi-policy	×	×	✓	✓	✓
Vector reward	×	×	✓	✓	✓
Pros	It performs faster while training the network using random mini-batches from temporary memory instead of recent transitions	It performs better to reduce observed overestimations compared to DQN learning	Learn parallel multiple policies and adapt rapidly when there is a change	It acquires multiple unsupported policies in a single run by constructing a tree-walk using upper confidence bounds (UCB)	Finds all deterministic PF policies for an MORL settings
Cons	Overestimation and the agent follows greedy approaches to fix the Q value, it often may lead to less optimized policy and increase time complexity	Single stream double DQN performs worse than Dueling DQN (DuDQN)	It may be stuck in local optima due to the assumption of optimal policy in advance	The agent does not support bootstrap and has to wait for the outcome to update the predictable reward	Requires high convergence time
Used in a dynamic environment	×	×	×	×	×

In the context of the dynamic environment and handling MOMDP, the algorithms are considered which support multi-policy (i.e. Multi-policy DQN, MO-MCTS, and MPQ). To understand the working procedure of the proposed algorithm, it is important to describe how policy selection works in the given context. The following section reinvigorated this discussion.

### 5.3 Policy Search in DQN

Usually, policy search methods aim to find out the optimal policies based on gradient-free or gradient-based approaches (Ng, 2003; Felix Yu, 2017). However, several successful methods in DRL disdained the commonly used backpropagation method in the literature in favour of evolutionary algorithms that are predominantly gradient-free based policy search algorithms (Gomez and Schmidhuber, 2005; Koutník et al., 2013). However, in the context of RL, a combination with an ANN weight are used to train large networks; such a technique resulted in the first deep neural network to learn the RL task, straight from high-dimensional visual inputs (Koutník et al., 2013). Recent work has reignited the interest of using policy search methods for RL as they can potentially be distributed at larger scales than techniques that rely on gradients (Salimans et al., 2017). For example, Figure 5.1 shows the Space Invaders game (Bellemare et al., 2012) where the agent can learn what parts of the image is important (Mnih et al., 2015).

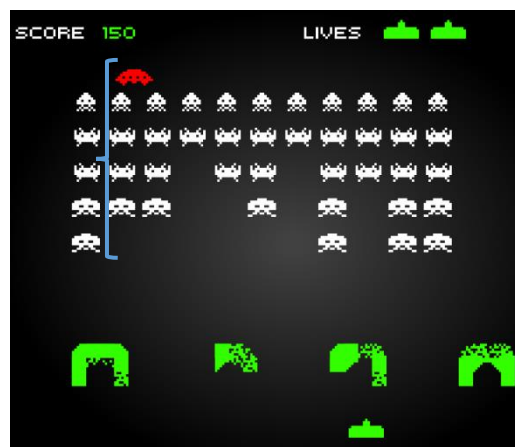


Figure 5. 1: Importance sampling based on the image in the Space Invaders game

However, the main strength of the DRL has remained the backpropagation. In general, the RL rule helps to determine the networks to learn stochastic policies by applying a trial-error procedure in a task-dependent manner. This allows the agent to find out where to look at an image to track and decide to capture the action information. In these scenarios, the stochastic variable helps to fix the coordinates (x, y) based on the pixels to crop an image and hence, determine the coordinates

of a small crop of that image. Therefore, the computational complexities are reduced. This procedure in the domain of RL is one of the most popular methods to track the optimum policy (Mnih et al., 2013).

However, searching directly for a policy represented by a neural network with many parameters can be a problematic approach and the model can suffer to stick in a local optima (Hauser, Eftekhari and Matzinger, 2018; Mark Lake, 2001). The possible solution to solve this problem is by using a guided policy search (GPS) (Levine and Koltun, 2013). In this procedure, a few sequences of actions from another controller learn by implementing supervised learning in a combination with the importance sampling. This sampling is responsible for the off-policy models. Thus, this mechanism effectively biases the search to find out the optimal policy. In addition, this procedure iterates to find out the optimal policies by sampling trajectories and optimising the trajectory distributions.

On the other hand, to track the changing policy the agent needs to balance the exploration towards unexplored areas of the state space and the exploitation of non-dominated actions. However, while traversing the whole environment, there might be some points where the agent needs to apply its own experience and already learned policy. The process of applying this policy is different in the context of single and multi-objective environments for the RL agent. In the single-objective environment, the learned policy can be easily traced by applying a function that calculates each action where it will find the highest value for each state by applying greedy policy selection.

Contrarily, in the context of a multi-objective environment where the agent needs to handle multi-policy simultaneously, then a modified definition of a greedy policy needs to be defined for MORL by applying vector rewards to select the actions (Vamplew et al., 2011). In the multi-policy selection environment, the agent needs to find out the actions consistently in order to retrieve the desired policy based on the  $\overrightarrow{Q}(s, a)$ . A question may arise – what the steps would be if the agent does not or cannot work in this way? There is a high chance that the agent

would be stuck in a local optimum because the obtained Pareto front is based on the local information and there would be no guarantee that the agent may find out the solution which is reached to global optima or at least close to the global optima.

To solve this behaviour of the agent and to get rid of this problem, a global greedy policy needs to be defined that continuously follows or tracks a given expected return vector. In other words, the agent needs to find out the global policy  $\pi^*$  from the initial state to the terminal state. Nevertheless, this process is not a straightforward way due to stochastic behaviour and the dynamics of the environment (van Wissen et al., 2012). Therefore, it is important to track the necessary actions for the desired vector rewards. In the selected test cases, for this study, the agent needs to traverse the environment to find the optimal policy. Consequently, the agent starts from a starting point and after that, it follows a particular policy based on the vector reward to achieve the highest expected reward i.e.,  $V\pi^*(s)$ . This accumulated reward is updated to the Q table at the end of each episode. For each action  $a$ , the agent averaged the immediate reward  $R(s, a)$  and the long-term discounted reward  $\gamma R(s, a)$ .

However, the selection is only executed if the sum of these two rewards is equal to the target vector that needs to be followed. After achieving the target vector, the agent proceeds to the next state and observe the corresponding action on that particular state (Barrett and Narayanan, 2008). When the vectors have not completely converged or the transition become stochastic, the action is selected based on the average value of the experiences, which is stored in the replay memory. In the experiments, a dynamic value has been assigned that helps to find out the optimal Q value based on the state and action. Thus, the optimal policy is dynamically selected for both test cases. The following section is going to discuss the way of selecting a meta-policy based on the policy search mechanism which has been discussed here.

## 5.4 Meta-policy Selection Mechanism

As far as the dynamic environment is concerned, a meta-policy (i.e. governing the policies in the policy lifecycle) has been introduced that defines which policy needs to be counted and prioritises the objectives. However, it is difficult to define the meta-policy with respect to all the objectives. In other words, all the objectives can be fully satisfied only in utopia. Therefore, in reality, the optimisation algorithm can achieve only a compromising solution which best fits for a particular situation according to the preferences. To overcome this problem and achieve the best compromising solutions, a parity value (i.e. dynamic weight) has been introduced before summing up the Q-values from DQNs. To make the system robust and personalised, a preference value or bias value (e.g. if any) has been added to prioritise the alternatives and matches the criteria in a dynamic Pareto optimal set  $PS(t)^*$ . Moreover, the agent helps to devise the action selection mechanism that can lead to a better decision. The meta-policy is selected by updating the process of selecting each action by choosing  $Q(s, a)$  as follows:

- a. The set of  $Q(s, a)$  is updated with vectors from  $S'$  at time step  $t$  for the first time after performing an action  $a$  in state  $S$ . In the meantime, actions in  $S'$  are sampled at the time  $(t + 1)$  that is previously unexplored.
- b. As estimations are being done over time, it is likely that the  $Q(s, a)$  can be created, updated and deleted at time step  $(t + 1)$ . As a result, vectors, in  $S'$  can be dominated by other vectors (Moffaert and Nowé, 2014).

However, in the dynamic environment, the location of the optimum moves either deterministically or stochastically. This move can also be linear, non-linear, periodical or random over time during the optimisation process. The core difference between stationary and dynamic optimisation problems is that any component that exists in the environment changes over time in the latter case. In the defined problem with time-varying, the desired goal of the algorithm is:

- a. to find the global optima to detect the changes and
- b. tracking the changing optima over time.

To identify the meta-policy in the changing environment the followings need to be done:

- a. detecting changes by re-evaluating detectors or changing components,
- b. detecting changes based on the algorithmic behaviours and
- c. balancing the objectives using objective relation mapping based on a dynamic weight (parity value).

In an MOMDP, where the dynamics of the problem changes to different trajectories through the metric space  $\delta(t) \in \Delta$ , are the parameters of the function  $t$  can be expressed by Equation 5.1.

$$\delta(t) \rightarrow \delta(t+1) \rightarrow \delta(t+2) \rightarrow \dots \quad (5.1)$$

Where,  $T$  is a period index such that  $(t) \neq \delta(t+1), \forall T$

The following Figure 5.2 demonstrates the Q value mappings for the different policies in DQN structures. For the visualisation purpose, the selected policies have been colour-coded whereas policy is chosen based on the yellow columns of different objectives. Eventually, the MORL agent produces the green one which is the mapped version of the meta-policy.

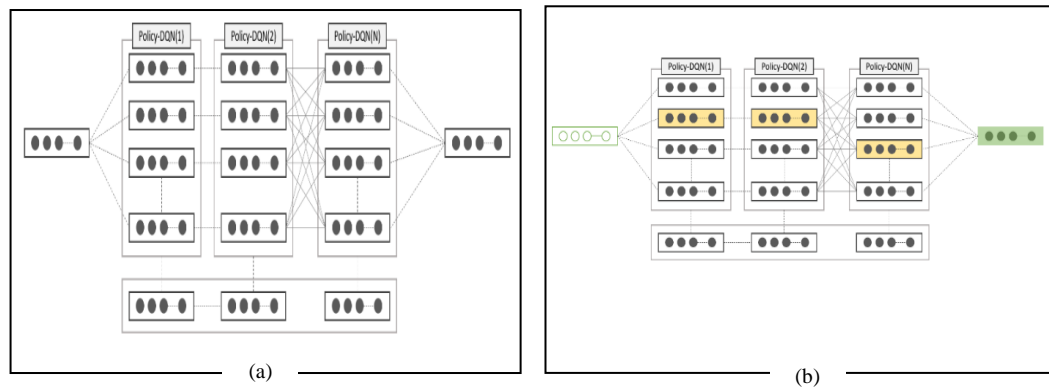


Figure 5. 2: A visualisation of Q value mapping in DQN architecture for governing policies to detect the changes of the optima (a) without mapped (b) mapping among different objectives

The meta-policy of selecting a policy is defined by the parity value which is obtained by the objective-relation mapping (ORM) in a dynamic environment to satisfy or closely satisfy the objectives. From Figure 5.2, it is also observable that the policy is selected based on the individual input of the DQNs which are the representatives of the state and action for a particular objective.

The agent selects an action  $a$  in the state  $s$  to form the vector of  $\vec{Q}(s, a)$  that is composed with the reward vectors  $\vec{r}_t = [\vec{r}_1, \vec{r}_2, \dots, \vec{r}_n]$ . For each pair of the policy  $\Pi(i) = \vec{Q}_i(s, a)$  and the next policy  $\Pi(i + 1) = \vec{Q}_{i+1}(s', a')$ , the agent creates a matrix as in Equation 5.2:

$$\Pi_{n[i],n[i+1]}(\theta(t)), \text{ where: } \theta(t) = \text{Dynamic changes } (\theta(t - 1)) \dots \dots \dots (5.2)$$

A transformation matrix is obtained by the following Equation 5.3:

$$\Pi_{transform} = \Pi_{n[1],n[2]}(\theta(t)) \cdot \Pi_{n[3],n[4]}(\theta(t)) \dots \Pi_{n[p-1],n[p]}(\theta(t)) \dots \dots \dots (5.3)$$

Update the Pareto Front (PF) position by the following Equation 5.4:

$$X(t + 1) = X(t) \cdot \Pi_{transform} \dots \dots \dots (5.4)$$

Where the changing severity  $\Delta f = |T(PF) - O(PF)|$  is set to different points based on true PF  $T(PF)$  and obtained PF  $O(PF)$ .

The following Figure 5.3 shows the relation mapping for different objectives based on selected policies where the convolutional layers direct to form a compromised solution. This process is predominantly executed by the neural network of the deep layer and by adjusting the weights and bias for each neuron (Schmidhuber, 2015).

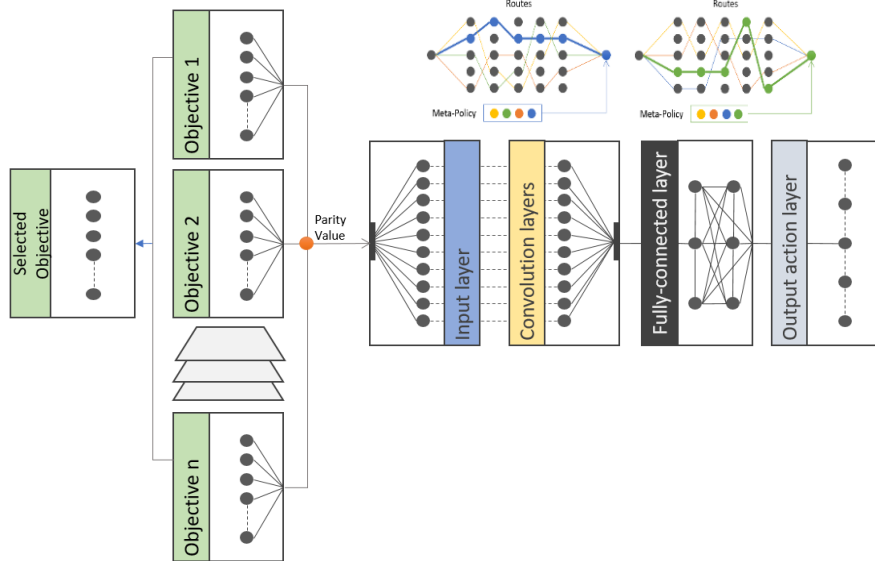


Figure 5. 3: Object-relation mapping to find out the equilibrium between objectives in a PODON architecture

From the above Figure 5.3, it is observable that the agent determines the compromising solutions based on the balance of the several objectives. These Q values are forwarded by the DQNs which consist of the set of state and action values for a particular episode that is generated by the emulator. In the deep layer, the weights of the neural network are adjusted based on the backpropagation procedure (Baldi and Sadowski, 2018). The obtained Q value by the agent is the average value from the DQNs that characterise all the objectives. Therefore, the selected objectives are the representation of the compound structure of all the objectives. Needless to mention, this value is made by the Q values in a finite horizon. In other words, this can be represented as the most compromising solutions that the agent could achieve in a particular episode.

The selection of these average accumulated Q values of each objective is sent to a buffer which follows the FIFO (i.e. first in first out) mechanism. This ensures the dominated solutions has removed or replaced by the non-dominated solutions based on the associated vector reward that estimates  $Q_{n-1}(s, a)$ . Therefore, the agent has now represented a non-dominated policy unless it is replaced by a new one. In other words, the agent finds the optimal policy which is good at least in one objective and not worst in any other objectives (Lwin, Qu and Kendall, 2014).

Figure 5.4 shows the changing policy in a dynamic DST environment (silver only for simplification) that is governed by the agent. The highlighted arrows (e.g. yellow and red) denote the changing path where the red one (i.e. treasure value 74.0) deserves more attention than the orange one (i.e. treasure value 50.0). The directions are changed to explore the greater value with the aim to maximise the highest expected reward. Since the objectives can be added with a separate DQN structure, thus, this method provides the robustness and its capacity to scale up.

Therefore, a new objective can be added or there is an opportunity to add several objectives based on the requirements. In view of that, the proposed algorithm proves that it is not rigid to accommodate new objectives and produce the solutions without fine-tuning the agent. The agent finds the best solutions which are not



biased considering the current state due to the random allocation of the experiences that are stored in the replay memory and not only depends on the current transition based on the actions. Accordingly, the agent governs the policy that is compromising among all the objectives. However, it is possible that in a particular episode, the agent may not achieve the best optimal policy if the parity value is influenced by the preference values. Thus, the ultimate selection of the policy is designated. The colour code of Figure 5.4 shows the emphasising attitude of the agent while traversing the environment. The deep red colour shows the higher intensity for choosing that node compared to the orange one and the arrowhead directs the movement towards north-south or east-west performed by the agent.

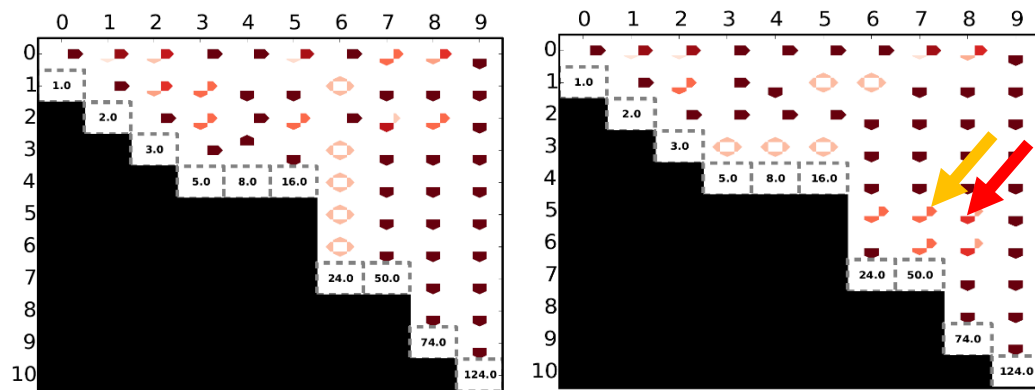


Figure 5. 4: Meta-policy selection in the dynamic DST environment (silver only)

The following Figure 5.5 shows an episode where the treasure values change randomly but with the set of the fixed values of silver and gold. Figure 5.5 also shows the true Pareto frontier (i.e. non-convex) achieved by the agent. See Appendix A for the full set of the reward distribution for every environment and the Pareto frontier values. The true Pareto frontier achieved by the agent is consists of three gold and seven silver treasures.

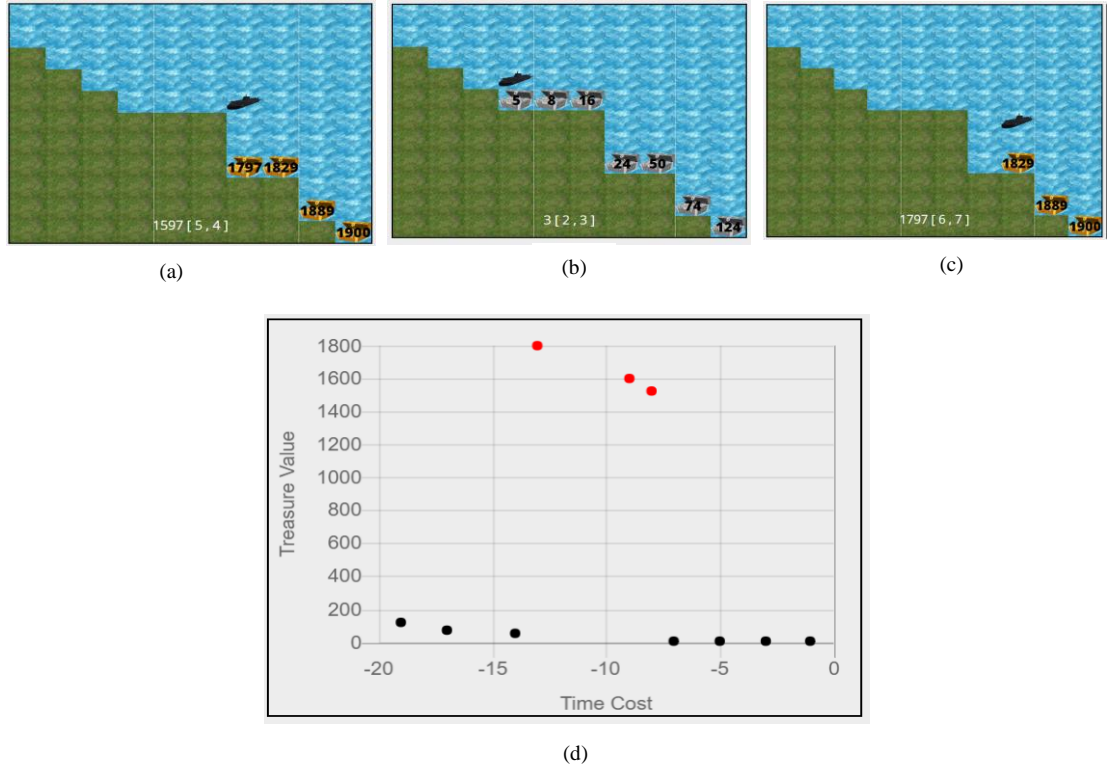


Figure 5. 5: The agent's traversing (a, b and c) in the dynamic DST (silver and gold) at the different states (d) true PF

## 5.5 Tracking the Optimal Policy

In the proposed algorithm, the goal of the agent is to find out the optimum policy by maximising the expected reward that is usually defined by the sum of the total rewards. Here, the interaction between the agent and environment goes in the finite set of the states. Therefore, the tasks are episodic and within a finite horizon. The agent needs to find out the policy which will determine the actions. In this case, a policy depends only on the outcome of each DQNs according to the objective and time. Since the policy is utilised to communicate within the environment will not be the same that is already learnt. Therefore, only one policy can be followed per episode. However, several policies still need to be improved concurrently considering the next state  $S'$  at time  $t+1$ .

Notice that PQDQN learns the vector in  $Q(s, a)$  for each possible combination of the non-dominated vectors. However, only the subset of non-dominated state vectors of  $V(s)$  can be determined to retain action vectors in other  $Q$  sets by Bellman's optimality principle (Martin, 2011). Thus, the proposed algorithm traces the number of policies.

The following Figure 5.6 shows the decision space (a), objective space (b) and tracking the optima in slow (c) and fast-moving (d) situations over time in a dynamic environment which is observed by detecting the moving optima (Lam, Branke and Abbass, 2005; Farina, Deb and Amato, 2004).

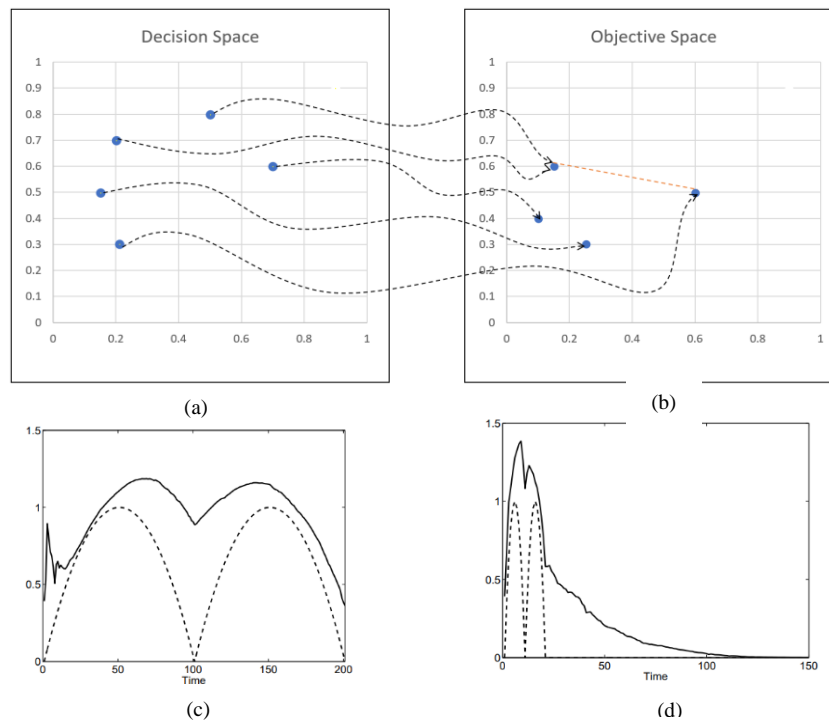


Figure 5. 6: Objective space and the tracking of global optima: (a) decision space (b) objective space (c) tracking slow-moving optima and (d) tracking fast-moving optima

Since the PQDQN algorithm is based on the Q learning, it is also an off-policy algorithm that means learning a new policy is not identical as like as the learned one. In addition, the agent learns several policies at once, which is the most important feature needed for the multi-objective scenario. This feature allows the agent to follow any policy which has been learned during the interaction within the environment.

Let us consider an MOMDP where an agent needs to move from the state  $S_0$ , source node to a destination node  $S_1$  with the transition probability 0.5 and the discount factor is 1. This leads to an expected accumulated reward of (0.5, 1) for this multi-objective MDP in RL settings. However, if we consider the MOMDP as depicted in Figure 5.7 where the agent needs to go across from the source node  $S_0$  to the destination nodes either  $S_7$  or  $S_8$ . In this case, the agent needs to traverse all the nodes to reach the terminal states. Since the task is episodic and the traverse starts at  $S_0$  and it ends whenever the agent reaches the terminal state, in this case, both  $S_7$  and  $S_8$ . Hence, the responsible action  $a_1$  has a probabilistic outcome and may lead to states  $S_1$  or  $S_2$ . Similarly, with the actions ( $a_2$  and  $a_3$ ), the agent can reach either  $S_3$  or  $S_4$  along with the reward functions  $\vec{r}_3 = (r_e, r_f)$  and  $\vec{r}_4 = (r_g, r_h)$  and the associated discount factor(s)  $\gamma$ , respectively. Later, the agent may reach the terminal state  $S_7$  either following the  $\vec{r}_7$  or  $\vec{r}_8$  vectors to maximise the highest expected rewards. Similarly, to reach the terminal state,  $S_8$ , the agent follows similar ways. In this case, the agent utilises the reward functions  $\vec{r}_2 = (r_c, r_d)$ ,  $\vec{r}_6 = (r_k, r_l)$ , and  $\vec{r}_9 = (r_q, r_r)$  and the discount factor up to  $\gamma^2$ . Let us also consider that for the all terminal states  $S_7$  and  $S_8$ , we have  $V(s') = (0, 0)$  with the reward  $r = \{0, 0\}$ , i.e., each state has a single zero vector at the initial state. Therefore, there will not be any action at the terminal states.

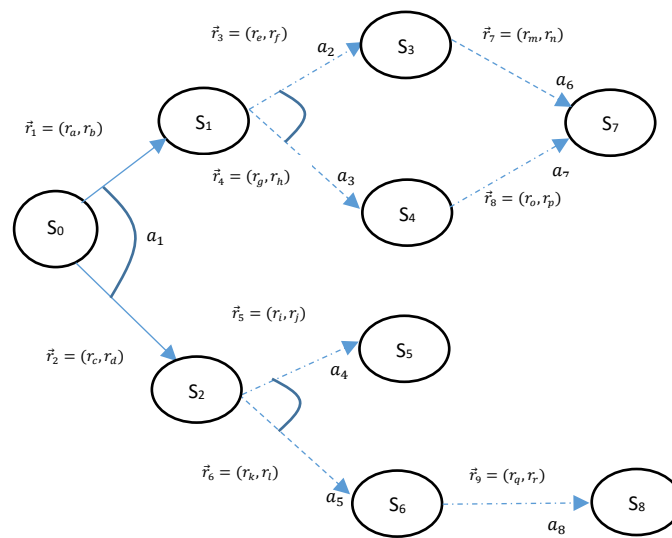


Figure 5. 7: Sample state transition between different states along with the vector rewards

Table 5.2 shows a general view of gaining the rewards by the agent with only eight states along with the sample vector reward of  $\vec{R} = \{r_x, r_y\}$  where x and y represent magnitude and direction respectively. Here, in the initial state, the agent receives the value function of  $\vec{v}_0 = [(r_a, r_b), (s_1, a_1)]$ . Similarly, in the later episodes, the agent gains reward based on the Q value estimations.

Table 5. 2: Progression of vector function estimation based on the Q values over time with the PQDQN

Time	State	Q (s, a)	$\vec{V}_t = (s_n)$
0	$S_0$	$(s_1, a_1)$	$V(s_0) = \vec{v}_0 = \{[(r_a, r_b), (s_1, a_1)]\}$
	$S_0$	$(s_2, a_1)$	$V(s_0) = \vec{v}_1 = \{[(r_c, r_d), (s_2, a_1)]\}$
1	$S_1$	$(s_3, a_2)$	$V(s_1) = \vec{v}_2 = \{[(r_a, r_b), (s_1, a_1)] + \gamma [(r_e, r_f), (s_3, a_2)]\}$
	$S_1$	$(s_4, a_3)$	$V(s_1) = \vec{v}_3 = \{[(r_a, r_b), (s_1, a_1)] + \gamma [(r_g, r_h), (s_4, a_3)]\}$
2	$S_2$	$(s_5, a_4)$	$V(s_2) = \vec{v}_4 = \{[(r_c, r_d), (s_2, a_1)] + \gamma [(r_i, r_j), (s_5, a_4)]\}$
	$S_2$	$(s_6, a_5)$	$V(s_2) = \vec{v}_5 = \{[(r_c, r_d), (s_2, a_1)] + \gamma [(r_k, r_l), (s_6, a_5)]\}$
3	$S_3$	$(s_7, a_6)$	$V(s_3) = \vec{v}_6 = \{[(r_a, r_b), (s_1, a_1)] + \gamma [(r_e, r_f), (s_3, a_2)] + \gamma^2 [(r_m, r_n), (s_7, a_6)]\}$
4	$S_4$	$(s_7, a_7)$	$V(s_4) = \vec{v}_7 = \{[(r_a, r_b), (s_1, a_1)] + \gamma [(r_g, r_h), (s_4, a_3)] + \gamma^2 [(r_o, r_p), (s_7, a_7)]\}$
5	$S_6$	$(s_8, a_8)$	$V(s_6) = \vec{v}_8 = \{[(r_c, r_d), (s_2, a_1)] + \gamma [(r_k, r_l), (s_6, a_5)] + \gamma^2 [(r_q, r_r), (s_8, a_8)]\}$

The following Figure 5.8 shows the time-changing property for one particular Pareto-optimal solution to another solution. For the visualisation purposes, time-unit has been shown in Figure 5.8 where it reflects the next time and the changing Pareto frontier based on a particular episode. For instance, in episode 0 and with the state  $(s_1, a_1)$ , the agent moves from one place to another when the arrow changes from 10 to 11. This change is traceable by  $\Delta t = (t_{11} - t_{10})$  based on the changing time step. In other words, this can be represented as the change tracker based on the time interval that traces any changes that may occur in the environment.

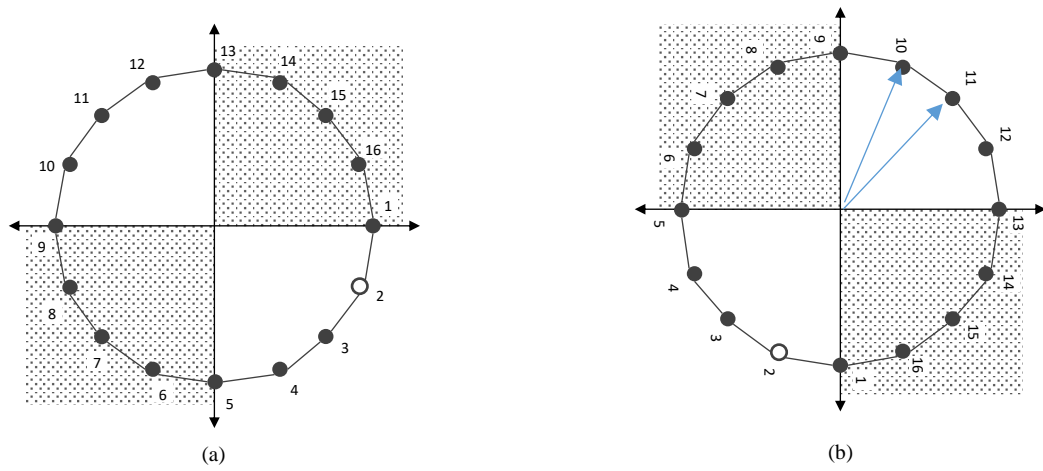


Figure 5.8: Visualisation of the changing time while the agent is traversing in the dynamic environments

The following Figure 5.9 represents the movement of the two submarines in the DST (attack by enemy) environment. The arrow indicates the points of clashes between the two submarines. Figure 5.9 (a) represents clashes between two submarines without the intensity (i.e. an accentuating attitude to choose a node) whereas Figure 5.9 (b) shows the clashes with the intensity of the different movements by the agent while traversing the environment.

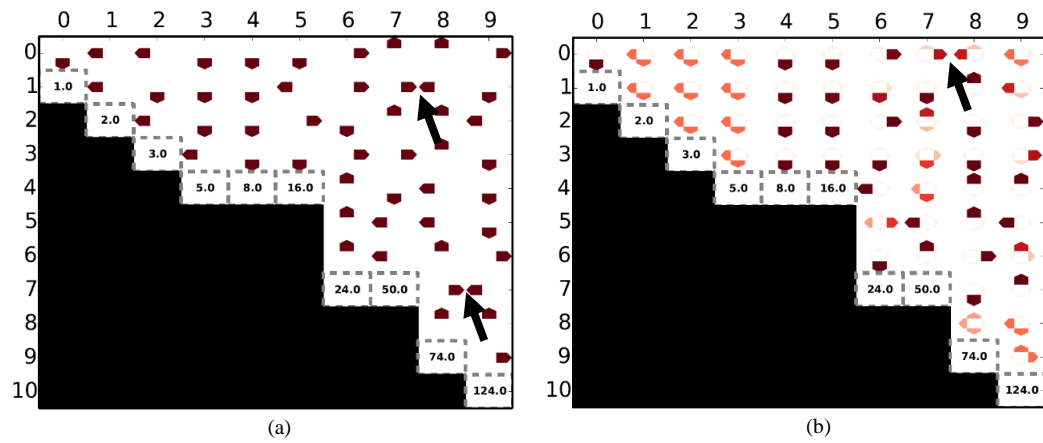


Figure 5.9: The agent's traversing (clashes) in the dynamic DST (attack by enemy) in different timestamps (a) without highlighting the intensity of the agent's move (b) with the intensity of the agent's move

Now, the Pareto frontier will be examined based on the changing time that is observed by the agent in Figure 5.10. In the first instance as shown in Figure 5.10,

when the agent traverse at timestamp 1, the agent has received  $(-1, 10, 0.01)$ , where,  $-1$  is the time cost, the health meter is pre-defined as  $10$ , and the treasure value is  $1$ . In this case for the consistency of the graph and the visualisation, the reward is divided by  $100$  and that is why it becomes  $0.01$ . In another case, when the agent moves from the timestamp  $2$  to  $3$ . The health condition is  $10$  at this timestamp which means that the agent is still not hit by the enemy submarine. Consequently, the agent achieves the treasure values which is  $2$  with the cost of the time penalty of  $-3$ .

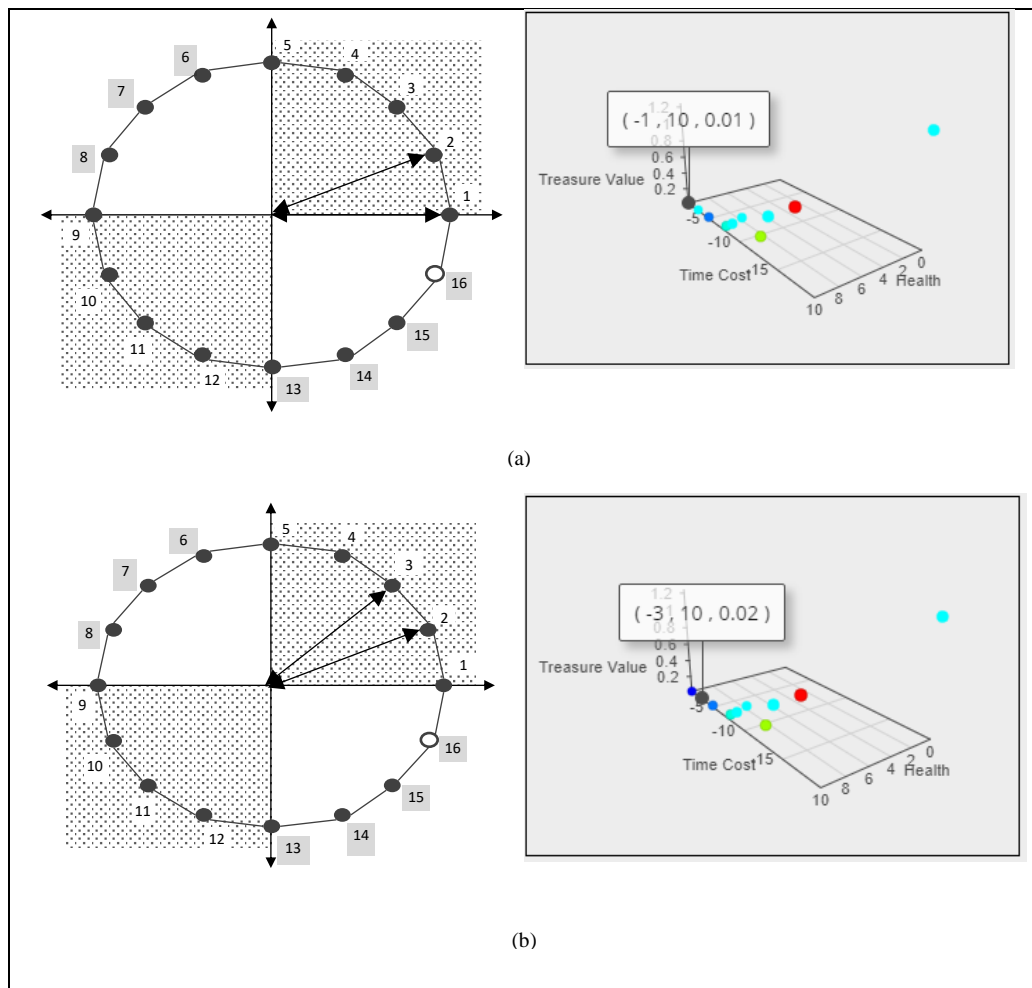


Figure 5.10: Changing Pareto frontier (a) at timestamp 1 to 2 (b) at timestamp 2 to 3

Moreover, the following Figure 5.11 illustrates the overall condition of the agent where the agent gets  $-2$  penalty at timestamp  $7$  to get the treasure value of  $50$ . This is the first time in this episode when the agent collides with the enemy submarine

and thus, the health meter is damaged, and the health condition decreases from 10 to 8. It also means that from the next state the agent has to be careful to save itself when it faces the enemy submarine.

Hence, the agent is now assigned with another new objective such as searching the treasures and save itself at the same time when it is attacked by the enemy submarine. Furthermore, when it comes to the 19<sup>th</sup> timestamp, the agent loses its whole health power and now it becomes 0 to achieve the highest reward 124 treasure value in that particular episode. In other words, the enemy submarine hits the agent 5 times before the agent has achieved the highest treasure in this episode. This also reflects that the developed benchmark is dynamic because of the changing behaviour at different timestamps.

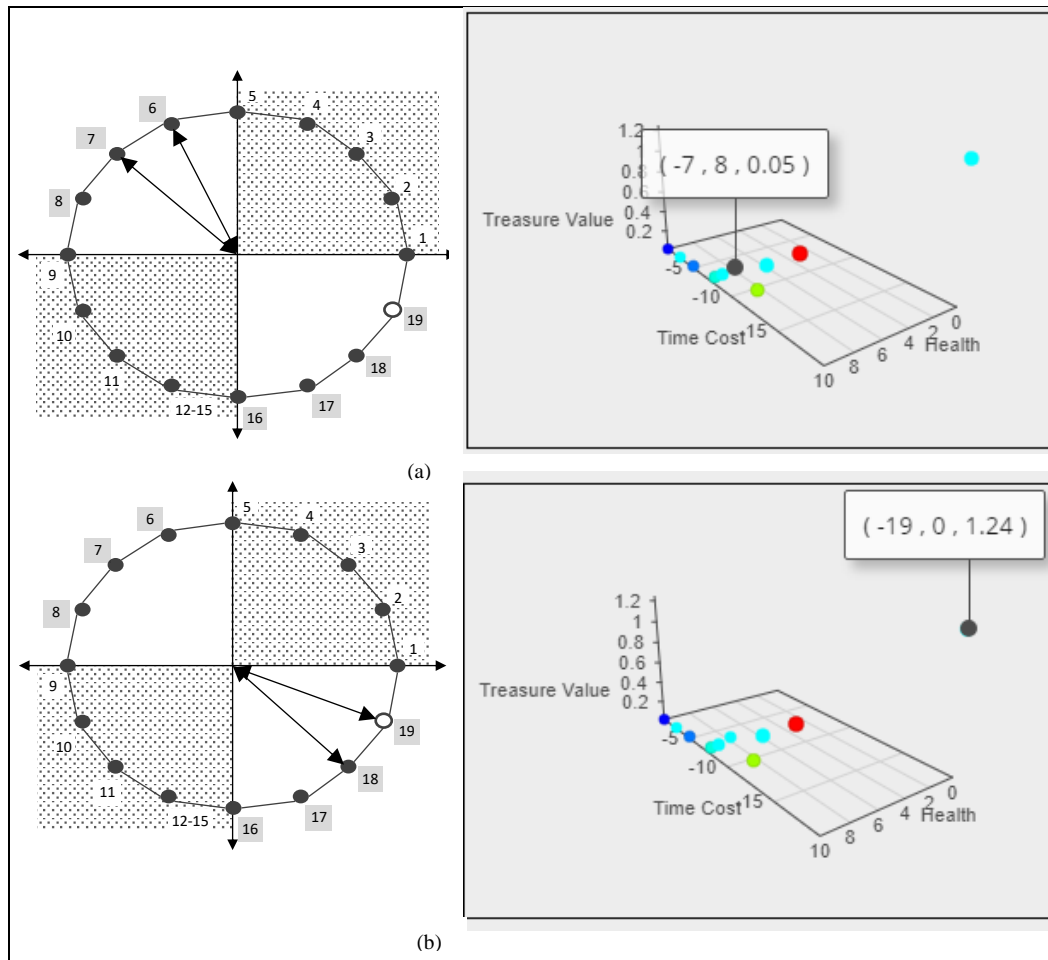


Figure 5. 11: Changing Pareto frontier (a) at 7<sup>th</sup> timestamp and (b) at 19<sup>th</sup> timestamp



The following Figure 5.12 shows a visualisation of the DST environment where the agent traverse based on the best and worst path using a meta-policy.

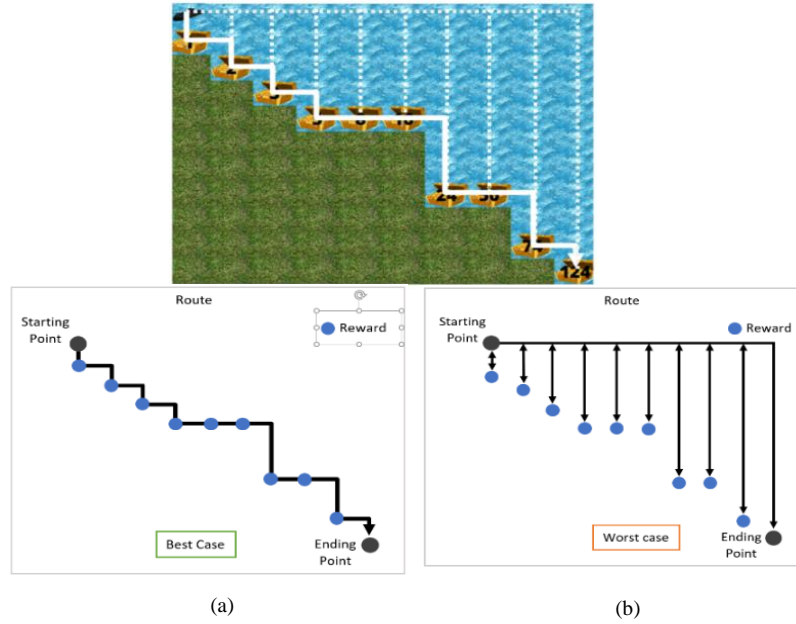


Figure 5. 12: A visualisation of the trajectory based on the meta-policy (a) the best-case scenario and (b) the worst-case scenario

It is worth mentioning that in a special scenario in a dynamic environment such as the agent creates a loop or the environment is influenced by unknown factors, the changes can be undetected, and thus, the moving optima cannot be identified. As a result, the solution cannot be reached.

## 5.6 High-level Architecture of the Proposed Algorithm

In this section, the fundamental architecture of the proposed algorithm has been described. RL is an influential paradigm for sequential decision-making under uncertainties (Kantas, 2009; Ravichandiran, 2018). On the other hand, MORL deals with the multi-objective possibly conflicting objectives. To solve this sort of problem using RL, one solution is to make the multi-objective problem into a single-objective problem and then apply the algorithm. Another solution is to reach the optimal solutions based on the Pareto dominance and Pareto frontier. Therefore, to solve the MOO in this thesis, a set of solutions that approximate the

real Pareto front has been considered (Deshpande, Watson and Canfield, 2013). This process can be illustrated in Figure 5.13.

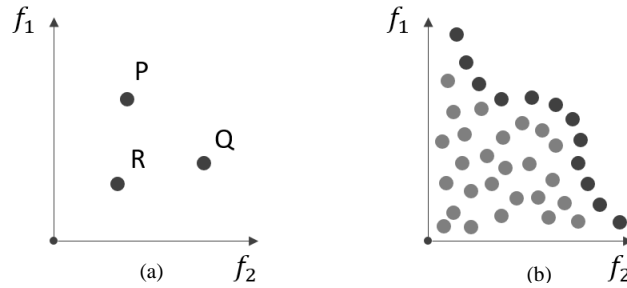


Figure 5. 13: Sample of the Pareto dominance (a) decision space (b) objective space

In the above Figure 5.13, the solution P is better than R in at least in one objective and P is also superior or at least equal to R in other objectives where  $f_1$  and  $f_2$  represent max and min functions respectively. In other words, the solution R is dominated by the solution P and Q. This scenario can be matched with the considered test cases where the MOO problem can be solved using the observation of the Pareto dominance. As it is difficult to find out all the Pareto frontier in test case 2 because of those are unknown based on the definition that has been set in this study, therefore, the best-known Pareto frontier has been chosen for test case 2.

To construct the algorithm in the MORL settings, as mentioned earlier, the reward functions should deal with the vectors. As a result, defining a vector was crucial. Here, a vector is an array of numbers and the space that the vector contains is called the vector space. A vector has a magnitude and direction. In these test cases, the vector is two dimensional and follows the matrix operation (Idris, 2015). For example, this matrix is formed of the rows and columns such as in the dynamic DST, which is a grid-world of  $10 \times 09$ .

Before going to further details, MORL architecture needs to be determined in the context of this study. This is crucial to characterise the basic construction of MORL in advance. At this point, the technique for the reward distribution needs to be settled. It is worth mentioning that the MORL structure is not identical as

like as the traditional RL architecture. The main reason is that handling more than one objective simultaneously makes MORL different from the single objective RL. In the MORL settings, the agent's present vector reward is delivered by the response signal of a specific state from the environment for each objective. As discussed in chapter 2, there are several approaches to solve MORL problems. In this thesis, the weighted sum approach has been used. In this approach, the optimal policy is picked up by the weighted value of the Q function. In addition, in the settings of MORL, the agent learned the reward functions by a vector operation such as summation or multiplication of the two-dimensional matrix. The action is chosen by applying an  $\epsilon$ -greedy selection mechanism, with the probability of  $\epsilon$  for a random and  $(1 - \epsilon)$  for an action that is selected (Ruiz-Montiel, Mandow and Pérez-de-la-Cruz, 2017). These vector rewards represent the corresponding actions that are selected for a particular state. Each objective has the threshold values subject to their constraints. The agent estimates the objective values based on these threshold values. In other words, the expected maximum and minimum values are the upper bound and the lower bound for a specific objective.

While traversing the considered environments, the agent has to decide the optimal policy that needs to be selected. In the first environment (dynamic DST), the agent navigates the environment and selects the actions to achieve the highest accumulative reward. This reward is based on the vector (i.e. magnitude and action) which is offered by the next states. The environment changes to a new state during traversing due to the dynamic nature of the problem. Because of the episodic environments, the Q value of the observed environment also gets changed for dynamic behaviour. Moreover, the environments in this thesis are considered based on the non-deterministic finite automata (NDFA) due to their dynamic behaviour over time (Becher, Carton and Heiber, 2015).

Additionally, a replay memory (RM) is used for training the DQN. The purpose of using a RM is to store the transitions which are observed by the agent. This also allows the agent to reuse the data at the later stage for traversing and finding the optimal policy. By random sampling, the transitions that build up a batch helps to

stabilise and improves the DQN training procedure. Therefore, the following classes are required:

- *Transition*- that represents a tuple of a single environment
- *ReplayMemory*- which is a cyclic buffer that stores recent transitions.

Now, considering the Q value approximation, the agent looks for the pair of the state and actions. The combination of the state and actions form a table that contains the Q values. In the simplest form, the Q value can be denoted by Equation 5.5:

$$Q[s, a] = (Instant + discounted) reward \dots\dots\dots (5.5)$$

Where, the agent gains the immediate reward when it moves from one state to another while performing an action. On the other hand, the discounted reward determines the future performance of the agent.

Generally, the Q values come up from the scenarios where the agent looks at the Q value to confirm which action to take or which policy needs to be implemented when in state S. In a particular state S, the agent needs to determine which action is the best. Before moving to the next state, the agent goes through all the possible actions so that it helps to determine the largest values of the action and thus, the agent selects a particular next state. For MDP, the policy that needs to be implemented depends on the current state (Silver, 2015; Mitchell, 2006) to maximise the rewards to get the optimal solution.

The proposed solution consists of two different blocks which represent a common RL setting such as selecting states and actions. The agent interacts within the environment and selects the actions based on the mapping with different objectives. This mapping is named objective relation mapping (ORM). The following Figure 5.14 shows the components of the proposed algorithm to handle dynamic MOMDP.

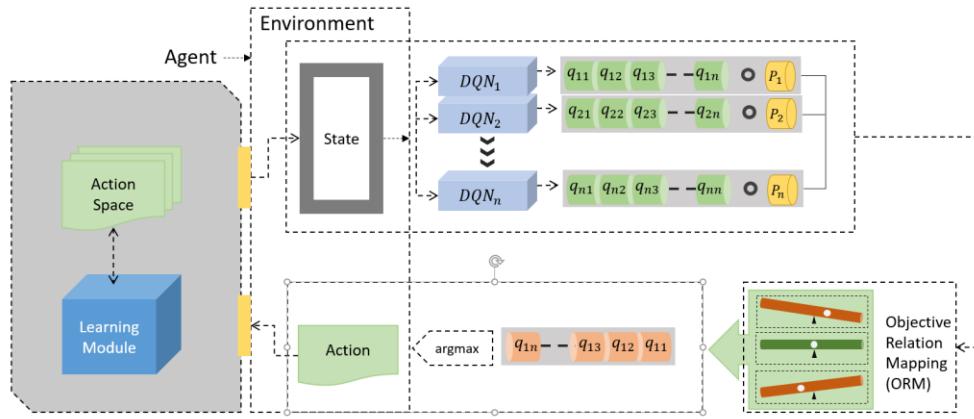


Figure 5. 14: High-level architecture of the proposed parity-q deep q network

From the above Figure 5.14, it is observable that the Q values are selected based on the states and actions. The state is selected based on the DQN networks and this value is sent to a stack where the agent looks for the best Q value. After that, the highest value is mapped with the ORM module where it forwards the best compromising objectives after satisfying the constraints. This module balances the Q values to achieve the possible optimal Q values for the different objectives by averaging them or multiplying with the preference values (e.g., if any). This action is performed based on the balancing of all the objectives. Therefore, the solutions that are provided by the PQDQN must be a compromising solution if there is more than one objective are available. From this module, the Q value forwards to a buffer where the final action gets selected and the agent has learnt the environment in an episodic style. In the buffer setting, the first Q values pass to the agent since it follows the queue mechanism. It is worth stating that the highest Q value passes because of the *argmax* operator. The learning module of the agent is based on the hyperparameters that are mentioned in Chapter 4. In addition, while traversing, the agent ensures that the traversing is completed to visit all the nodes so that there is no unexplored state. Thus, the agent interacts within the environment and learns the optimum values which lead to select the policy as discussed in the earlier section.

The target of the proposed algorithm is to detect the changes and then tracking the changing optima (i.e. local optima or ideally the global optima) over time. From the below algorithm 1, it is noticeable that the vector rewards need to be provided

for each action and prioritise the objectives (i.e. if needed as like as the DST-enemy attack environment to prioritise health more than treasure). Therefore, unlike the other DST environments, DST (attack by enemy) belongs to three objectives. After that, the agent needs to convey the state-action pair into a deep Q network and get the highest achieved Q value (multiply with the preference value -if any) in the stack for each episode. Furthermore, the agent makes a map among the objectives with parity-Q value and scalarise the Q value to determine the non-dominated solutions (if any) and distance from each other. Finally, the agent sends the achieved Q value to a buffer for *argmax* and finally select the action and update the target network.

The following algorithm 1 shows the proposed PQDQN.

Parity-Q Deep Q learning Network (PQDQN):

```

Line 1: Initialise: Temp Memory T // temporary list
Line 2: Initialise: Set action value-function Q with random/arbitrary numbers
Line 3: Initialise: Set two fully connected layers for the nth objective (two/three different objectives) attached to the vector reward  $\vec{r}_{t+k+1}$  for multiple policies
Line 4: Initialise: Set the temp functions for each objective
Observe: initial state S
Line 5: for each episode  $e \in \{1, 2, \dots, M\}$  do
    Line 6: observe reward r and new state s'
    Line 7: if (state!=0){
    Line 8: select an action a arbitrarily with the possibility of  $\mu$ 
    Line 9: else
    Line 10: select action  $a = \text{Max}_{a'} Q(s, a')$  // identified the highest Q value to select the action
    Line 11:}
    Line 12: store experience in Temp T [s,a,r,s']
    Line 13: Get sample transitions from Temp T
    Line 14: Create a list/table R (to store combined Q values)
    Line 15: While (Transitions!=0 in store experience){

    Line 16: If (ss'== terminal state)
        Line 17: tt= rr+yMaxa· Q(ss', aa')
    Line 18: else
        Line 19: set tt=rr
    Line 20: Train the Q network with pre-defined loss
    Line 21: s=s'
    Line 22: end
    Line 23: Create a stack for objective 1,  $\hat{S}_1$  which consists of  $[E_1 \dots E_n]$  for the value functions
    Line 24: while ( $\hat{S}_1$  is not empty)
    Line 25: Push in stack  $\hat{S}_1 \rightarrow E_1: \{Q_1(s_1, a_1) \text{ value of Objective}_{1(\vec{r}_{t+k+1})}\}$ 
    Line 26: If ( $E_1 < E_n$ )
    Line 27: Pop the stack  $\hat{S}_1$ 
    Line 28: Insert  $E_n$  into the stack
    Line 29: Otherwise, Insert  $E_1$  into the stack
    Line 30: end
    Line 31: Create a stack for objective 2,  $\hat{S}_2$  which consists of  $[E_1 \dots E_n]$  for the value functions
    Line 32: while ( $\hat{S}_2$  is not empty)
    Line 33: Push in stack  $\hat{S}_2 \rightarrow E_2: \{Q_2(s_1, a_1) \text{ value of Objective}_{2(\vec{r}_{t+k+1})}\}$ 
    Line 34: If ( $E_2 < E_n$ )
    Line 35: Pop the stack  $\hat{S}_2$ 
    Line 36: Insert  $E_n$  into the stack
    Line 37: Otherwise, Insert  $E_2$  into the stack
    Line 38: end [continue the push/pop block up to nth objectives]
Line 39: Select the action <association map ( $E_1 \sim E_2$ ) || ( $E_1 \sim E_n$ )> based on the dynamic weight to balance the objectives (i.e. parity value)
Line 40: Store the value of  $Q = \overline{Q}_n^*(s, a)$  in a Buffer for enqueue and dequeue at time t (i.e. changing optima).
Line 41: Select the argmax (Q)
Line 42: Update target network  $Q^* \rightarrow \text{optimum value among objective}_1, \text{objective}_2 \text{ and objective}_n$ 

```

This is to be mentioned that the internal architecture of the algorithm is based on the DQN and the basic principle of the stack and queue (Cormen, 2009) have been utilised in the context of data structure.

It can be summarised that a state is selected based on the probabilistic values of the actions. In a simplified way this can be classified as follows:

- a. Find the optimal policy,  $\pi^*(a|s) = P[A_{t=1}|S_{t=s}]$ .
- b. Decide where to go such as the direction based on the vector reward.
- c. Find the stochastic matrix.
- d. Once the policy is selected, the agent determines the next move.

In a simplified version, the traversing can be visualised by the following Figure 5.15 according to the above-mentioned arguments where the agent looks for the state and the corresponding actions. These are determined based on the vector rewards for the next state. Since the treasure value is 50 which is greater than 24 in Figure 5.15, the agent moves towards the higher value rather than visiting the 24 value in this particular situation.

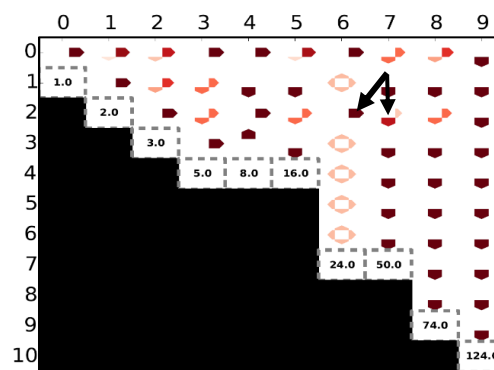


Figure 5. 15: Visualisation of the importance sampling while traversing in the dynamic DST

In a nutshell, the simplified approach of the PQDQN can be represented by the following Figure 5.16 in the context of Q value update. In the pre-defined MOMDP, when the agent traverses the environment, it tries to maximise the expected rewards. In both test cases, the agent updates the target network by updating the Q table. This is completed by relating the changes in the following stages in a particular timestamp and finally, the agent makes the decision by the proposed PQDQN algorithm.

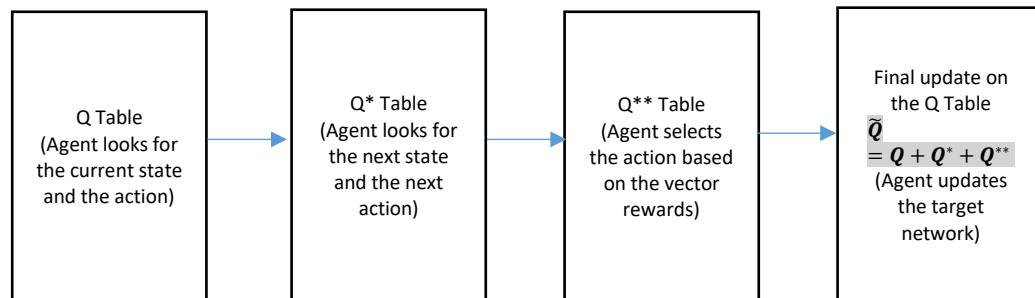


Figure 5. 16: Block-diagram for updating the Q value

## 5.7 Setup and Training the Model

Using deep learning by nature is CPU intensive, however, GPU is recommended to enhance the matrix operations.

The whole project has been implemented using the following tools:

For Development: MATLAB with AppDesigner Toolbox and Python libraries

Here, the system requirements have been mentioned so that one can also adhere to:

- a. A quad-core processor (Core i7-4770 CPU at 3.4 GHz) or higher (dual-core is not preferable)
- b. 16GB of RAM (8GB is not recommended)
- c. Bus Speed- 5GT/s DMI2
- d. At least 256GB of storage (512GB is recommended)
- e. Premium graphics cards can enhance the performance (otherwise performance can be hampered)
- f. Hyperparameters of the trained model can be found in Chapter 4.



The major python libraries that are used in this study is Keras and TensorFlow. Both are open-source libraries with a strong and active community. In the context of the Keras, it is a front-end library that is backed by Theano, CNTK or TensorFlow. However, TensorFlow backend is used in this study. The reason for using the Keras is that it is well-suited with the RL model and comparatively easy to formalise the activation functions such as using the sequential model offered by the Keras library.

On the other hand, in the context of the TensorFlow, a tensor is a multidimensional array of numbers. Here, vectors and matrices can be treated as 1-D and 2-D tensors. In the deep layer networks, tensors are mostly used for storing and processing data. Here, an episode can be represented using MDP and stored in a three-dimensional tensor. These dimensions can be represented as a matrix. Furthermore, Keras library is used based on the following considerations:

- This can be easily deployed for fast prototyping.
- Keras supports both the convolutional and recurrent networks regarding the deep layer architecture.
- It also executes seamlessly with the Central Processing Unit (CPU) and Graphical Processing Unit (GPU).

It is worth mentioning that the whole experiments are done based on the Keras model which is namely '*sequential*'. Besides, a graphical user interface (GUI) of the expert system (ES) has been developed to predict the vulnerable zones which are demonstrated in Appendix F.

## 5.8 Summary

In this chapter, the proposed algorithm has been discussed. This chapter deals with the working procedure of the agent and how it becomes proficient in an unknown environment. A step by step process has been demonstrated to formalising the object relation mapping (ORM). In summary, this chapter has got the following outcomes:

- a. In the PQDQN, the ORM works to balance the objectives to find the best compromising solution that is closed to the Pareto front.

- b. A comparison of the multi-policy DQN networks has been shown.
- c. Tracking the moving optima has been exemplified.
- d. Necessary tools and setups are also described.

All in all, this chapter provides an overall idea of using the PQDQN algorithm in the context of dynamic multi-objective environments as shown in Figure 5.17. In other words, this chapter provides the fundamental concepts of using the proposed algorithm in MORL settings. The tools with libraries are also discussed along with the Q value functions update which is performed by the agent.

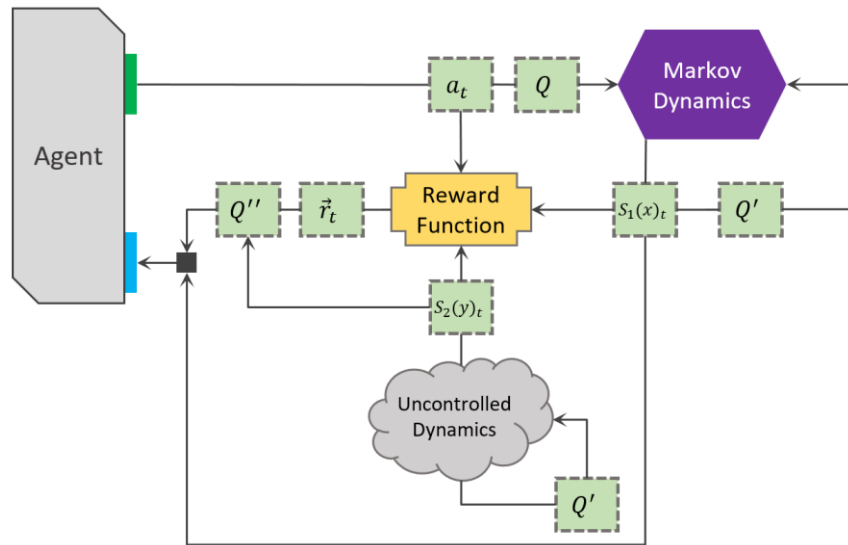


Figure 5. 17: Proposed PQDQN in the dynamic environment

The main advantages of using MORL can be summarised from this chapter as follows:

- The performance is enhanced with the highly diverse Pareto-optimal models such as in a dynamic environment while dealing with multi-objective environments.
- A trade-off can be determined between correctness and interpretability for sequential decision-making tasks.
- The agent remains flexible to decide about goals after learning.
- Constraints can be expressed by rewards “*being dominated by*”- that directs a partial order which is sufficient for many RL approaches.

The following chapter will discuss the outcome of the considered algorithms.

## Chapter 6

### Results and Discussions

#### 6.1 Introduction

In this chapter, the findings and outcomes of this study have been discussed. There is always a need to analyse and assess the outcome to measure the performance of the different algorithms to evaluate the quality of the achieved solutions. These solutions are based on the convergence, diversity, and closeness to the obtained Pareto frontier (Cheng, Shi and Qin, 2012). Measuring different criteria are useful to compare and rank the effectiveness of different algorithms. Therefore, the choice of suitable measures and the statistical test is vital to produce a non-discriminatory judgment for the considered algorithms (Okabe, Yaochu Jin and Sendhoff, 2003).

For the performance measure, there are different approaches based on the problems' types. In the context of the static problems, the usual evaluation procedure of the obtained result is to check convergence at the end of the search process. However, this is not identical when it comes to a dynamic problem (Cámara, Ortega and de Toro, 2009). The crucial factor for the dynamic environment is that the effectiveness of the algorithm which does not only depends on the final outcome but also to its behaviour and robustness towards the dynamics (Helbig and Engelbrecht, 2013a; c).

Stereotypically, this process includes how the algorithms are able to detect the changes in the dynamic environment and acts accordingly. When the global optima move from one place to another, ideally the algorithm should be able to track the solutions as they move from one search space to a different search space to avoid the escaping tendency to look for the global optima (Morales-Enciso and Branke, 2014). In addition, in the context of RL settings, the solution of one state may look like the algorithm has reached the solution based on the local information. In this type of problems, the agent often gets stuck in the local optima if the agent follows

the greedy approach (Samonji and Watanabe, 2017). Consequently, the agent cannot reach the solution to detect global optima. Therefore, the agent must need to explore all possible states in the finite horizon to establish the compromising solutions based on the optimal PF and PS where the solution is better at least in one objective and not worst compared to any other objectives (Moritz et al., 2014). Hence, the agent needs to keep tracking the actions and based on that; it selects the policy over the changing environment. In this chapter, at first, the evaluation criteria have been set. After that, performance evaluation and statistical analysis have been explained broadly. Finally, the strengths and limitations of the proposed algorithm have been described.

## **6.2 Evaluating Criteria**

Since both test cases are defined for the first time, there are no known solutions or any previous references that can be used to validate the obtained result. Therefore, technically speaking, the best-known true PF and PS have been considered for two test cases as a reference point.

Another assumption made for the sake of the performance evaluation is that all the considered algorithms can produce a smaller value of a robust metric which reflects less sensitivity to perturbation. These assumptions were necessary to consider since there is a lack of dynamic MOO study in the context of reinforcement learning.

### **6.2.1 Evaluating criteria for the proposed benchmark**

In order to evaluate the benchmark, the categorisation of the benchmarks that depends on the dynamics has been taken into consideration. As mentioned in chapter 4, there are four types of dynamics based on the PF and PS such as type I, II, III and IV (Farina, Deb and Amato, 2004) as described in Chapter 4. This well-known scale is utilised to set the dynamics for the benchmark based on PF and PS. As a result, to make an impartial judgement of the existing and the proposed benchmark, a comparison between two benchmarks have been taken into

consideration. This criterion has been achieved by setting up the environment as described in Chapter 3. This comparison is also helpful to assess how the benchmark is satisfying the criteria in the dynamic environment.

Besides, visual perception can be made by watching the video as mentioned in Appendix A. In this video, it is observable that the environment becomes dynamic by the changing values and parameters in the gameplay.

### 6.2.2 Evaluation criteria for the considered environments

In order to do an impartial analysis, the first criterion is selected in the context of accuracy performance measures is generational distance (GD). Equation 6.1 defines the distance between the optimal PF and the true one. To calculate GD, knowledge of POF is required and a reference set of POF. In this study, the best-known POF has been considered for all the defined problems such as DST (silver and gold), DST (attack by enemy) and to predict the vulnerable zones. Additionally, scaling and normalisation of the objectives are required to calculate the GD. It is to be noted that GD is computationally expensive, especially where a large number of objectives are used (Helbig and Engelbrecht, 2013).

$$GD = \frac{\sqrt{\sum_{i=1}^{\eta_{POF^*}} d_i^2}}{\eta_{POF^*}} \dots\dots\dots (6.1)$$

Where,  $\eta_{POF^*}$  represents the number of solutions in  $POF^*$  (i.e. a set of scattered points of optimal PF) and  $d_i$  represents the Euclidean distance that places in the objective space. This solution also represents the distance between the obtained POF and the nearest neighbour of  $POF^*$ .

As discussed in Chapter 2, the second criterion is selected as inverted generation distance (IGD) which is introduced by (Sierra & Coello, 2005) to overcome the non-adherence by GD. It is defined mathematically by Equation 6.2:

$$IGD (PF, PF^*) = \frac{\sum_{v \in PF^*} d(v, PF)}{|PF^*|} \dots\dots\dots (6.2)$$

Furthermore, IGD is well-suited with monotony, because this measure is going to rate a POF with more non-dominated solutions that are close to POF as a better set than another POF that only has one solution which falls within POF. However, IGD is also expensive in terms of computational aspects. The distance function is utilised for the purpose of the scaling and normalisation of the objective function values as like as the GD.

Hypervolume (HV) is selected as the third criterion to measure the performance of all considered algorithms. It is utilised to measure the dominance of the solutions by a non-dominated set. HV can be determined by the following Equation 6.3:

$$HV = volume \left( U_{i=1}^S v_i \right) \dots \dots \dots (6.3)$$

Here,  $U_{i=1}^S$  represents the union of all hypercubes of  $v_i$  in accordance with a reference point of POF where  $i \in S$  (Cámara, Ortega and de Toro, 2009).

It is inevitable that the performance of the algorithms cannot be measured adequately by one unary metric. Therefore, to analyse the effectiveness of the considered algorithms, a thorough study has been conducted as mentioned in Chapter 2.

Finally, a set of common characteristics have been considered to compute the performance such as required steps and obtained rewards in an identical setting to avoid any partiality. In addition to that, Student's T-test is also considered for the statistical analysis based on IGD which measures how far an obtained Pareto front from the true Pareto Front achieved by an algorithm.

### 6.3 Performance Evaluation

In the context of the dynamic settings of RL agent, the evaluation of the agent's learning procedure is challenging, because the changes in the states/actions are not known by the agent in advance. This changing distribution of a state-action pair makes it difficult for the agent to selecting a particular policy. In order to lessen these instabilities, performance measures are evaluated, and results are analysed in comparison to each other. A number of experiments have been performed to test the overall evaluation. In most cases, the performance is evaluated based on the 100 runs by the agents.

Therefore, this section represents the comparisons in the context of the proposed benchmark, the training time of the algorithm, accuracy, error rate, performance measure (i.e. GD, IGD and HV) and statistical test. It is to be noted that one algorithm can perform better for a particular metric of the performance measure while it may not produce a good result in different settings. As a result, in this thesis, the comparison of the considered algorithms has been measured separately with respect to similar settings and measuring parameters. In addition, since the deep layer works as a black box, data visualisation techniques have been used to get meaningful insights into the performance for different algorithms. This visualisation is based on the TensorBoard which has been used as a primary tool for the analysis in this thesis.

In order to assess the performance of the considered algorithms that solve DMOPs, performance metrics are frequently used. One of the primary goals of these metrics is to assess the ability of how well the optimisation algorithms can track the changing Pareto front. Considering the performance measure of the proposed algorithm and a fair comparison among the considered algorithms, the comparisons are made with the well-known deep Q learning algorithms which can handle multi-policies such as multi-policy DQN (MPDQN), multi-objective Monte Carlo tree search (MO-MCTS), and multi Pareto Q learning (MPQ) as discussed in Chapter 5.

### 6.3.1 Comparison of the existing and the proposed benchmark

It is observable from Figure 6.1 that there are no changing parameters and constraints or changing objectives while traversing the grid-world by the agent in the existing DST testbed. In other words, the existing DST testbed has fixed PF and PS (Perez, Samothrakis and Lucas, 2013). While the proposed dynamic DSTs as shown in Figure 6.1 (b, c and d) have the changing parameters (i.e. DST – silver and gold) over time as well as the changing objectives (i.e. dynamic DST – attack by enemy). Therefore, the evaluating attributes have been set based on the variance of the PF and PS (Binois et al., 2015). Here, the agent hunts a treasure by traversing a particular state based on the vector rewards.

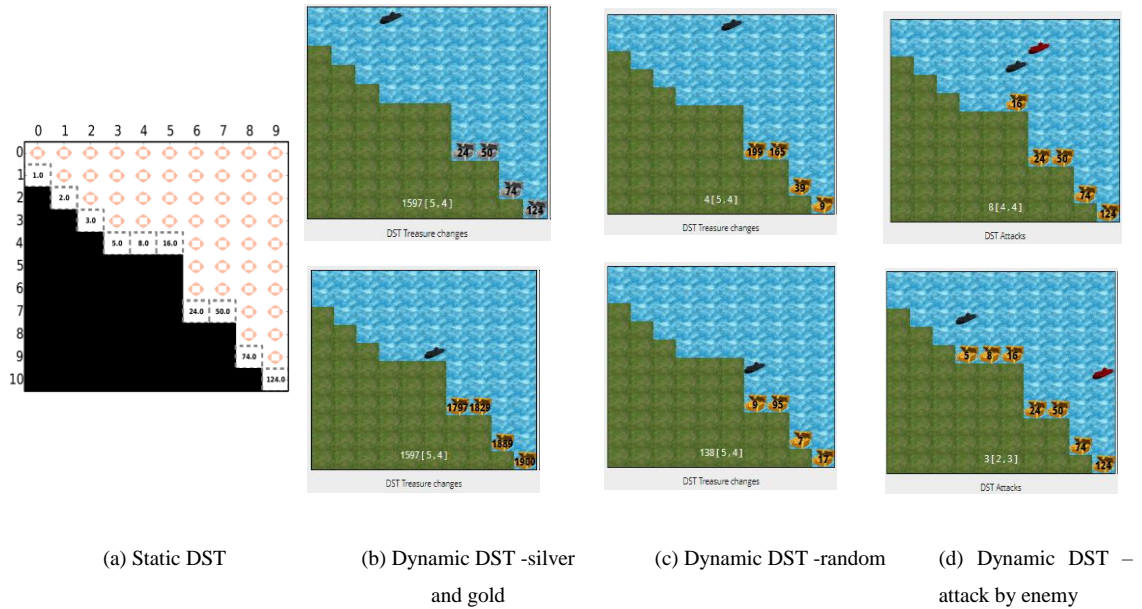


Figure 6. 1: All DST environments in this study

From the above Figure 6.1, it is also noticeable that the upper portion (i.e. treasure values) of Figure 6.1 (b, c, d) are changed from the lower part. For example, Figure 6.1 (b) shows the treasure values are getting changed based on the types of silver and gold. In the random environment, mentioned in Figure 6.1 (c), the values are changing randomly which has got no known patterns. Finally, the last environment as shown in Figure 6.1 (d), has an enemy submarine which continuously moving



and hitting the other submarine that damages the health meter of the agent. Hence, the process of attacking has affected the environment. As a result, this environment (i.e. attack by enemy) brings dynamics over time in terms of changing objectives while the agent needs to save itself in addition to hunting the treasures.

Besides, a new enemy submarine has been introduced in the third environment whereas the enemy agent's target is to attack the traversing submarine and damage its health condition. In this third scenario (i.e. dynamic DST-attack by the enemy), the agent needs to satisfy a new objective that dynamically changes while traversing. To sum up, the following Table 6.1 shows the changing PF and PS criteria of the existing DST and the proposed dynamic DSTs.

Table 6. 1: Comparisons of PF and PS between existing and proposed benchmark

<b>Testbeds</b>	<b>PF</b>	<b>PS</b>	<b>Objective(s)</b>
Classic DST	Invariant	Invariant	Invariant
Dynamic DST (random)	Variant	Variant	Invariant
Dynamic DST (silver and gold)	Variant	Invariant	Invariant
Dynamic DST (attack by enemy)	Invariant	Invariant	Variant

In this thesis, the study is being conducted based on the simulation where the RL agent needs to interact within the environment. Here, the conflicting objectives are handled based on the trade-offs between the objectives to find out the compromising solutions. This outcome is provided by the benchmark which has the pre-defined set of rules that supports the changing PF and PS as mentioned in the above Table 6.1. This is also responsible for ensuring the efficiency or fairness for the performed algorithms in such a dynamic environment.

Therefore, the proposed benchmark has satisfied the dynamics based on the changing optimal PF and PS. As a result, this benchmark follows the setting of dynamics as type II, III and IV which has been described in Chapter 4. Thus, the

proposed benchmark is successful in producing a dynamic multi-objective test bed in RL settings.

### 6.3.2 Comparison of the elapsed training steps and earned rewards

In this section, the elapsed time (i.e. number of steps) for the purpose of training has been discussed for all the considered algorithms for both test cases. The following Table 6.2 shows the average number of steps in thousands to reach the goal state and total expected return (i.e. reward) of MPQ, MO-MCTS, and MPDQN. The  $\pm$  shows the average higher and lower values to reach the goal state and obtain the rewards. The average calculation has been measured based on 100 agents.

Table 6. 2: Average number of steps and total expected return (in thousands) for MPQ, MO-MCTS and MPDQN

Cases	Environment	Metric	MPQ	MO-MCTS	MPDQN
Test case -1	Dynamic DST (Silver and Gold)	#Steps	34320.5 ( $\pm 253.1$ )	45065.2 ( $\pm 174.2$ )	58323.0 ( $\pm 223.6$ )
		#Return	6225.2 ( $\pm 123.6$ )	7055.0 ( $\pm 89.4$ )	5015.4 ( $\pm 100.6$ )
	Dynamic DST (Attack by enemy)	#Steps	76552.2 ( $\pm 348.6$ )	40349.5 ( $\pm 296.5$ )	68233.2 ( $\pm 463.8$ )
		#Return	2642.5 ( $\pm 229.0$ )	3405.6 ( $\pm 74.2$ )	2215.7 ( $\pm 152.2$ )
Test case -2	Water quality resilience	#Steps	93680.2 ( $\pm 328.1$ )	87462.3 ( $\pm 64.2$ )	94275.5 ( $\pm 325.9$ )
		#Return	22151.2 ( $\pm 213.5$ )	30255.3 ( $\pm 18.6$ )	14695.7 ( $\pm 175.8$ )

The following Table 6.3 shows the training steps until convergence over 100 agents for the proposed algorithm. In this table, the maximum steps have also been mentioned to get more insights into the proposed algorithm.

Table 6. 3: Average number of steps and total expected return (in thousands) for the proposed PQDQN

Cases	Environment	Parity Q Deep Q Network (PQDQN)		
		Average Steps	Maximum Steps	Average Return
Test case - 1	Dynamic DST (Silver and Gold)	25849.01	65882.19	9128.06
	Dynamic DST (Attack by enemy)	55894.24	92586.30	4305.21
Test case -2	Water quality resilience	86365.89	256748.53	38130.26

From the above two tables (6.2 and 6.3), it is clear that the proposed PQDQN earns highest expected rewards than MPDQN, MPQ and MO-MCTS in all the cases. However, the proposed algorithm takes reasonably higher steps compared to MO-MCTS in the Dynamic DST (attack by enemy) environment. The possible reason for this is the randomness generated by the enemy because of the enemy submarine hits the agent randomly and damages its health meter.

The following Figure 6.2 shows the heatmap of traversing by the agents in 4 different algorithms. The deep red zone shows the better path to be followed according to the true PF. For the simplification, here, the DST (silver) is only demonstrated.

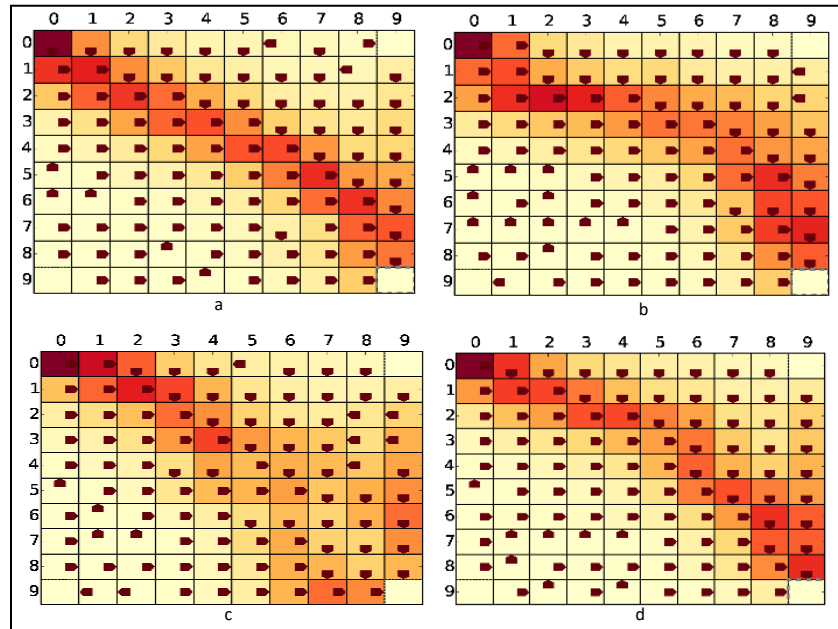


Figure 6. 2: Heatmap of average visited states over 100 agents for dynamic DST (silver only) (a) Parity-Q DQN, (b) MO-MCTS, (c) MPQ and (d) MPDQN

From the above Figure 6.2, it is noticeable that the average traverse of the 100 agents is more accurate for PQDQN compared to other algorithms.

The following Figure 6.3 shows the accuracy graph for dynamic DST (silver and gold) over 1 million steps where PQDQN performs better than other algorithms. In this environment, MPDQN performs worse than any other algorithms.

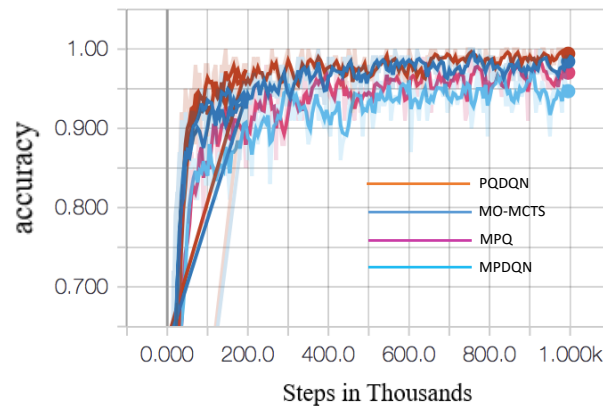


Figure 6. 3: Learning accuracy over 1M steps for dynamic DST (silver and gold)

The following Figure 6.4 shows the mean squared error of the different algorithms for the environment of the dynamic DST (attack by enemy) over 1Million steps where the developed algorithm performs reasonably better than the MO-MCTS algorithm.

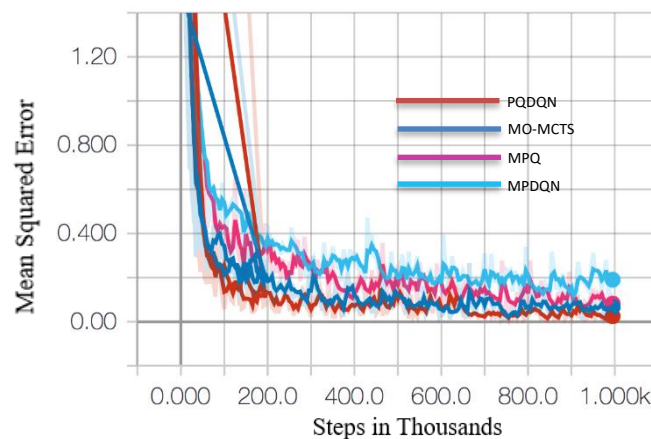


Figure 6. 4: Mean squared error over 1M steps for dynamic DST (attack by enemy)

Furthermore, the distribution of the weight-bias has been analysed. Considering a typical ANN, each neuron is connected via a weight to another neuron. Besides, the bias unit is an extra neuron to each pre-output layer that is not connected to the previous layers. The following Figure 6.5 shows the bias and weights distribution for the different convolutional and connected layers in the given network settings as mentioned in Chapter 4.

Figure 6.5 also represents the bias and weights in the convolutional network in the dynamic DST (silver and gold) environment with the learning rate of 0.001.

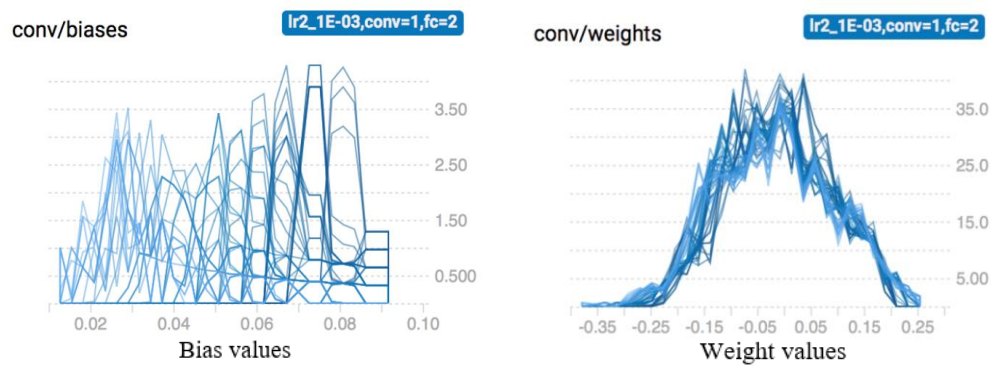


Figure 6. 5: Bias and weight distributions on the learning rate of 1E-03 for the dynamic DST (silver and gold)

The following Figure 6.6 represents the weight distributions in the Rectified Linear Units (ReLU) with the learning rate of 1E-03 for the convolutional network in the dynamic DST (attack by enemy) environment. In the case of the weight distribution, conv=2 has shown where it shows after 200K steps, the distribution of the weights becomes symmetric for the rest of the dispersal.

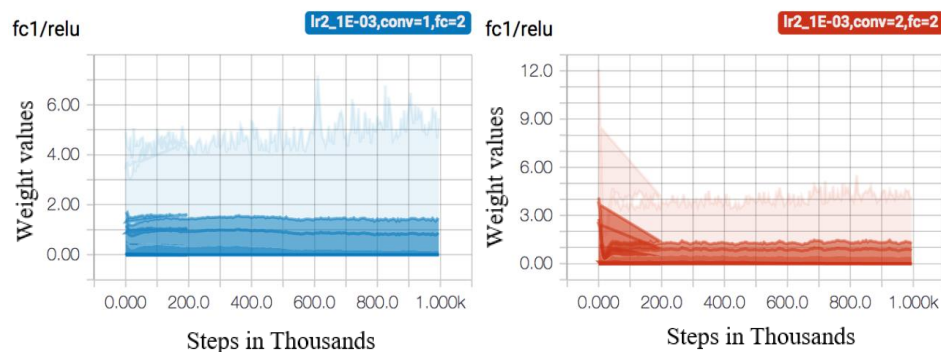


Figure 6. 6: Weight distributions on the learning rate of 1E-03 with conv =1 and 2 for the dynamic DST (attack by enemy)

The following Figure 6.7 represents bias and weight distributions with a convolution network and the fully-connected layer of 2 with the learning rate of  $1\text{E-}3$  for the prediction of the water quality resilience.

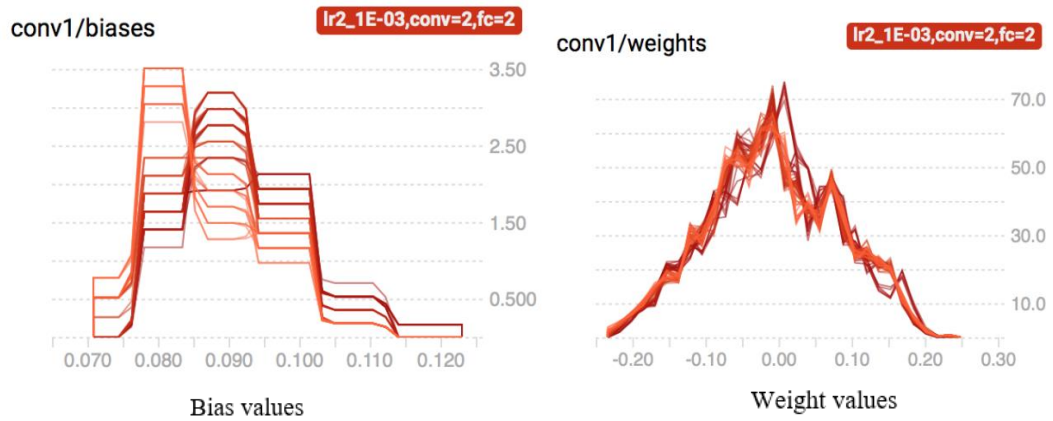


Figure 6. 7: Bias and weights distributions on the learning rate of  $1\text{E-}3$  for predicting water quality resilience

The distribution of bias and weights as shown in Figure 6.5, 6.6 and 6.7 reflect a normal distribution of the weights and bias values. This representation also illustrates that the overall connectivity of the neural network in the defined model is sparse and the network is robust to the noisy data. Hence, the solution to both test cases produced a robust solution that convergences with the limited number of iterations that have been set up in the network iterations for the sampling points and to find an accurate approximation of the true PF. A full list is mentioned in Appendix E.

Apart from the above criteria, as discussed earlier, a visualisation technique is used in this study to obtain more insights into the evaluation process and the relevant discussions. It is always important to visualise the high-dimensional data which can reveal unknown patterns and the perceptions of the dataset and the solution. For this reason, t-Distributed Stochastic Neighbor Embedding (t-SNE) and Principal Component Analysis (PCA) are used for the visualisation purpose (Laurens van der Maaten, 2018). This technique becomes one of the established features in the domain of ML and it is integrated with TensorBoard. However, it

is often argued that the interpretation from these two features are sometimes difficult, misleading and varies with the different input parameters (Wattenberg, Viégas and Johnson, 2016).

The purpose of using t-SNE is to produce an envisioning for non-linear and non-deterministic algorithms that reserves local neighbourhoods in the data. On the other hand, PCA tries to capture data variability in a few dimensions. The following Figure 6.8 and 6.9 show the PCA in a 3D view for the dynamic DST (silver and gold) and dynamic DST (attack by enemy) with the points of 3136 and the dimension is 1024 with 92% variance described of two objectives and three objectives respectively. From Figure 6.8, it is observable that the objectives are conflicting, and the obtained locations are almost in different positions on the map.

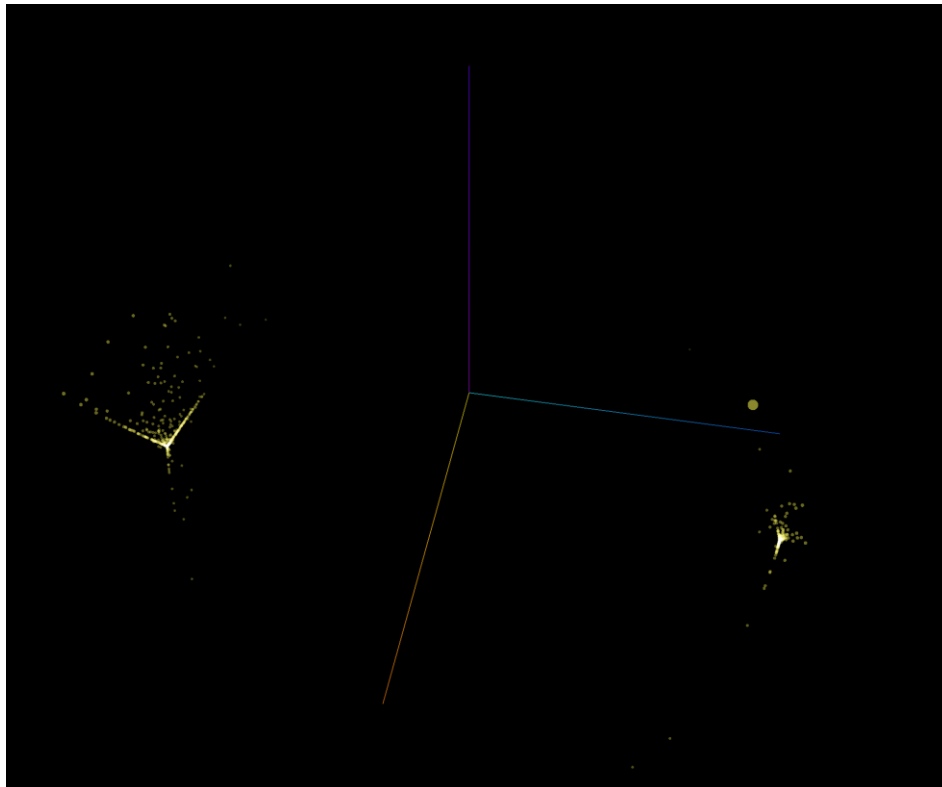


Figure 6. 8: PCA visualisation (night mode enabled) for the dynamic DST (silver and gold)

In the following Figure 6.9, it is also noticeable that the obtained objectives are scattered and loosely coupled in terms of the distribution of the objectives. The possible reason behind this is the randomness generated by the enemy submarine.

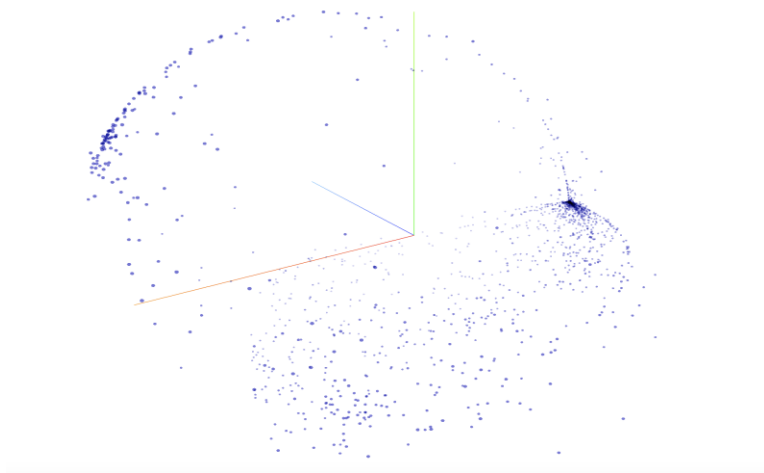


Figure 6. 9: PCA visualisation for the dynamic DST (attack by enemy)

In addition, the following Figure 6.10 shows the t-SNE with the perplexity (i.e. defined two the power of Shannon entropy) of 14, the learning rate of 10 and with 5000 iterations. The cluster represents the two different objectives, the lower portion is responsible with the highest value of the IQA and the upper cluster represents the combination of the objectives which is based on the lower values of the IET and IVA.

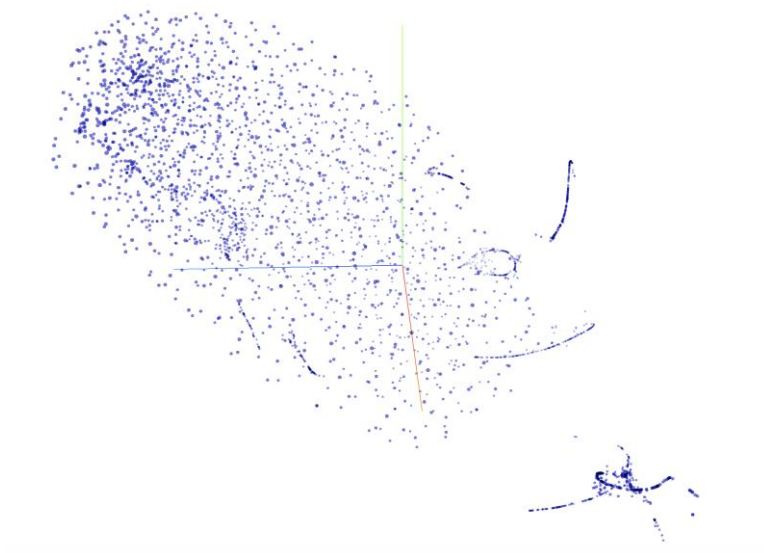


Figure 6. 10: t-SNE visualisation for the objectives in the WQR environment



### 6.3.3 Comparison of the performance of the different algorithms

In the dynamic environment, when algorithms solve the DMOPs, the environment changes frequently. As a result, the true PF found by the algorithms has gained certain outliers based on defined hyperparameters. These outliers are skewed the obtained results using:

- distance based performance by GD and IGD, which is the lower the better and
- the dominance is measured by the HV where a higher value of the HV is better (Lwin, Qu and Kendall, 2014; Zhou et al., 2019).

The effect of the outlier elucidates the calculation of the considered evaluating criteria as mentioned in section 6.2.2. Here, the performance comparisons are shown between four multi-policy DQN algorithms in terms of GD, IGD, and HV for the dynamic DST and the WQR environments. To gain an intuitive view of the considered algorithms, GD, IGD, and HV have been plotted for each of the cases where the results are averaged over 100 runs. The outcome confirms that all the considered algorithms are capable of converging.

For the dynamic DST (silver and gold), PQDQN performs better compared to the MO-MCTS, MPQ, and MPDQN with the smallest mean values for the IGD and GD metrics over 100 agents as shown in Figure 6.11. In addition, PQDQN achieved the highest mean values for HV that reflects its dominance in this testbed. In this scenario, the second-best algorithm is MO-MCTS that performs better than MPQ and MPDQN in terms of all the cases except for the IGD where MPDQN performs better than MPQ and MO-MCTS. However, MPDQN performs poorly in all the measured criteria, especially compared to the proposed algorithm.

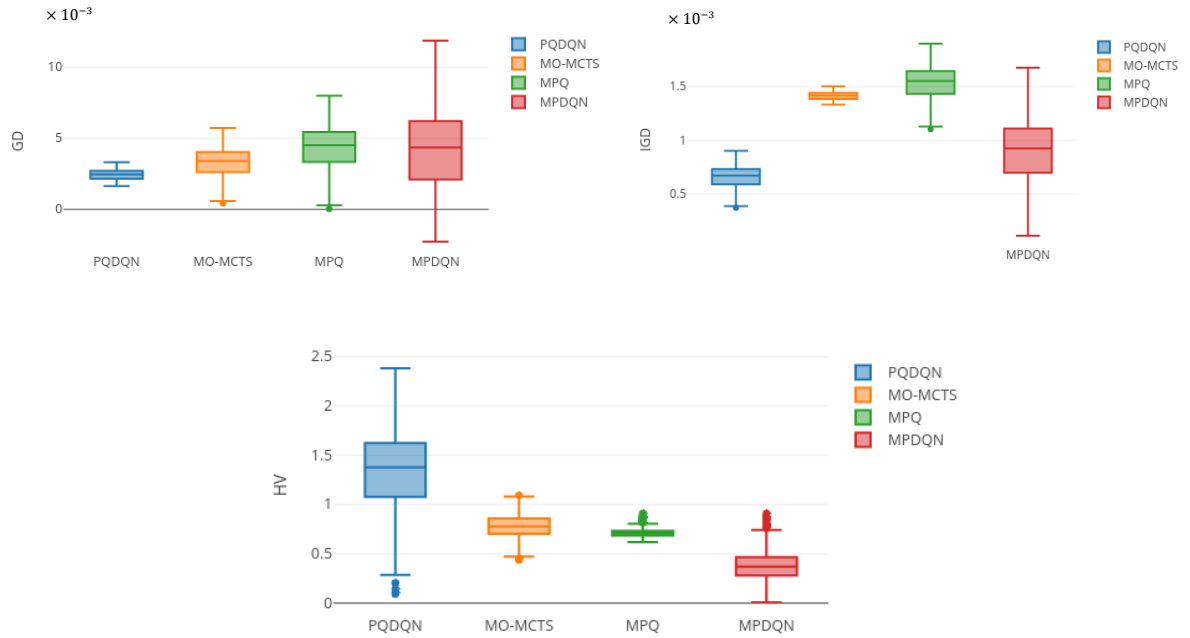


Figure 6. 11: Performance comparison of the algorithms in dynamic DST (silver and gold) in terms of GD, IGD and HV.

On the other hand, for the dynamic DST environment (attack by enemy), MO-MCTS performs better compared to PQDQN, MPQ and MPDQN for GD as shown in Figure 6.12. However, in terms of IGD, PQDQN has achieved a better result compared to all the considered algorithms. In this scenario, MPQ performs best in terms of HV and the second-best performer is MO-MCTS.

However, MPQ has performed poorly when it comes to IGD. As like as the other environment, MPDQN performs the worst except for the IGD. A possible reason of an unusual act of all the considered algorithms may be triggered by the randomness that is generated by the enemy submarine in this scenario that has been classified as type IV dynamics in the context of the DMOPs.

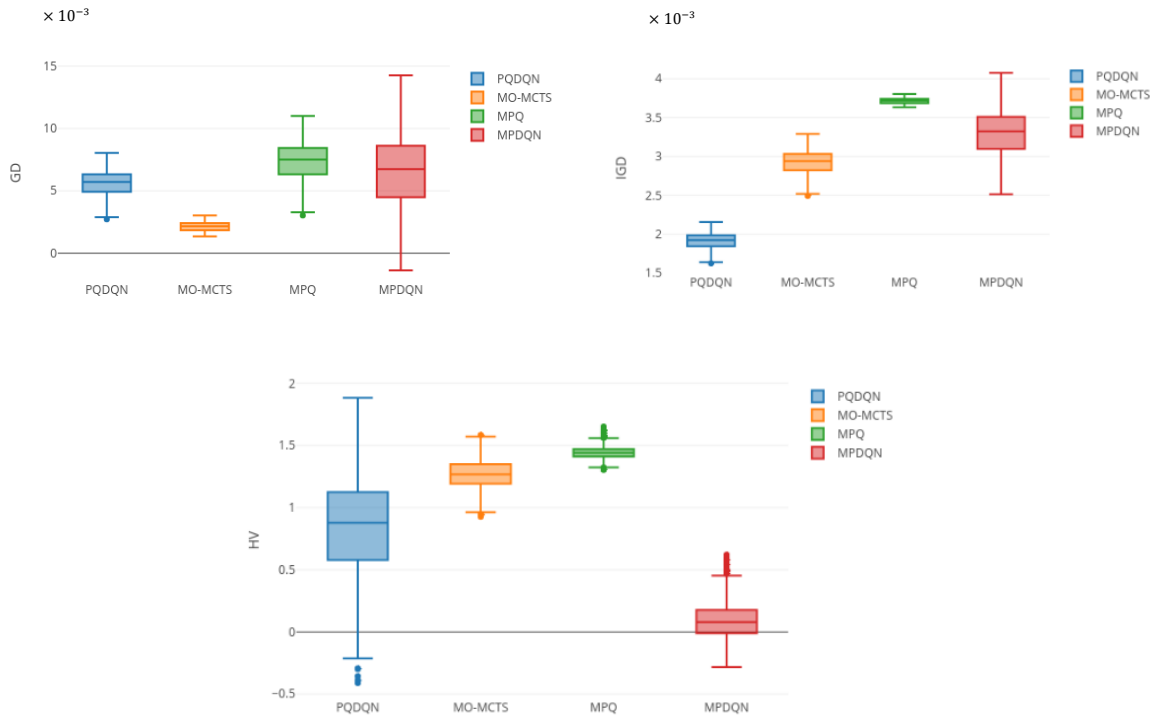


Figure 6. 12: Performance comparison of the algorithms in dynamic DST (attack by enemy) in terms of GD, IGD and HV.

The following Figure 6.13 shows the GD, IGD and HV measure to predict the vulnerable zones based on the water quality resilience environment. In this scenario, the proposed algorithm shows superiority in all the measured criteria (i.e. GD, IGD and HV). MO-MCTS becomes the second best-performer in terms of GD and IGD.

However, MPQ performs better compared to the MO-MCTS and MPDQN in terms of HV. As like as the other two dynamic environments, MPDQN has performed the poorest in terms of the GD criterion. Alike other two environments, MPDQN has performed better than MPQ in terms of IGD. However, MPDQN performed quite similar to MPQ considering the upper bound for HV in this scenario.

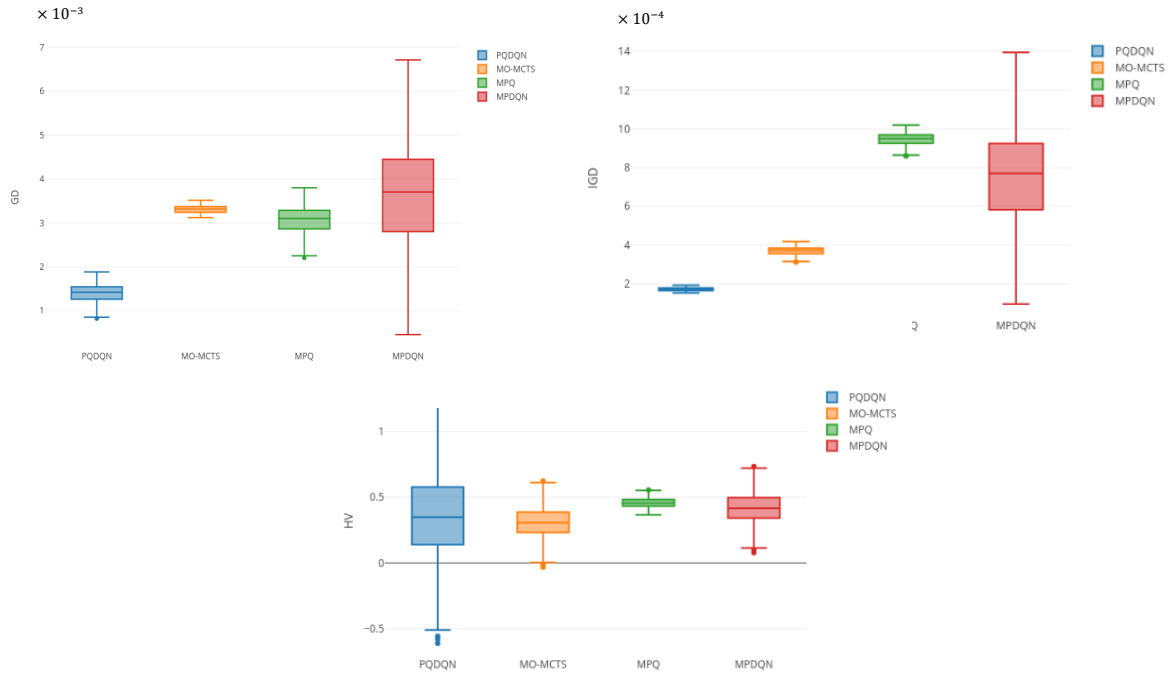


Figure 6. 13: Performance comparisons of the algorithms for WQR environment in terms of GD, IGD and HV

Overall, this can be concluded that the selected performance measuring criteria are satisfactory to judge the considered test cases (i.e. dynamic DSTs and WQR) and the performance of the algorithms. The results also show that the proposed algorithm PQQDN performs significantly better in all the test cases except the dynamic DST (attack by enemy) environment. PQQDN outperforms the considered algorithms in terms of convergence. The results also show that the 2<sup>nd</sup> best algorithm in this context is MO-MCTS.

However, MO-MCTS has performed poorly in some cases such as in the IGD calculation in the dynamic DST (silver and gold) and the HV computation for the WQR environment. Moreover, the MPQ has also performed well while it comes to the HV computation in almost every considered environment. In addition, the outcome also reveals that MPDQN performs poorly in every environment.

### 6.3.4 Comparison of the true PF for identifying the vulnerable zones

In this study, the proposed algorithm is responsible to identify the solutions that are closed to the Pareto front or a compromising solution that best satisfies the objectives. In the real-world scenario, the dataset has been classified based on the preset objectives (i.e. minimising IVA, IET and maximising IQA) as mentioned in Chapter 4. Therefore, the best-known true Pareto front has been established using WQI.

As a result, the obtained PF is compared with this best-known true Pareto front. The following Table 6.4 shows the vulnerable zones (e.g., 5, 6 and 15) which are predicted by the proposed algorithm.

It is noticeable that the proposed algorithm can identify all the vulnerable zones. Appendix F shows the full list of the most vulnerable stations based on WQI. It is also visible that, for the zones 6 and 15, the proposed algorithm can support all the solutions while for zone 5, it is short of 2 for the Pareto solutions.

Table 6. 4: Frontier features for the vulnerable zones

Predicted Vulnerable Zones	Points	Supported Solutions	Pareto Solutions
<b>5</b>	Ribeirão Quilombo	4	6
	Rio Jundiá		
	Rio Piracicaba		
	Rio Pirai		
<b>6</b>	Reservatório do	3	3
	Guarapiranga		
	Reservatório do Rio		
	Grande		
<b>15</b>	Rio Guaió	1	1
	Ribeirão do Marinho		

The following Figure 6.14 shows the comparison of the efficient Pareto frontier obtained by the PQDQN with the best-known Pareto frontier based on IQA, IET and IVA. Here, the red dots show the true Pareto front while the grey dots show the achieved Pareto frontier by the proposed algorithm.

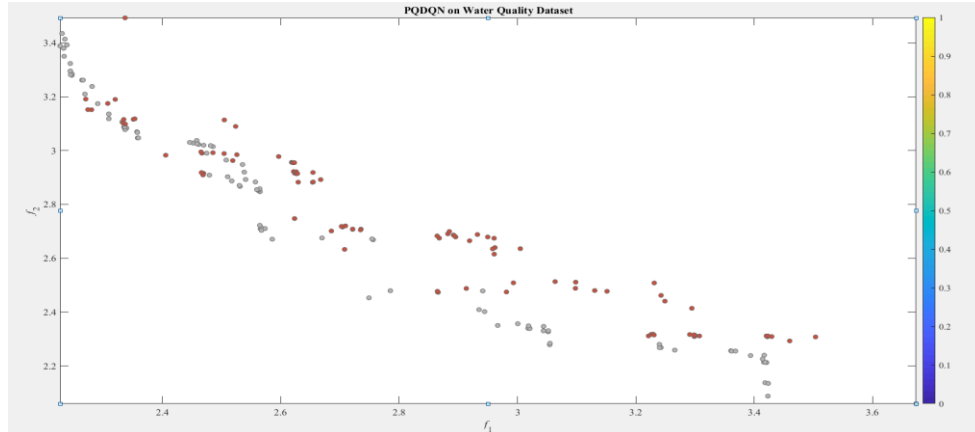


Figure 6. 14: Efficient frontiers; red dots: best-known Pareto front and grey dots: Obtained Pareto frontier by PQDQN

Figure 6.15 shows the intensity of the vulnerability (i.e. 1 equals most vulnerable) among zones where it is observable that 5, 6 and 15 are the most critical.

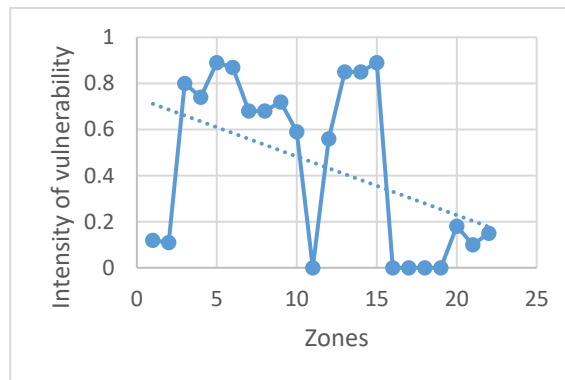


Figure 6. 15: Vulnerability in various zones

The predicted vulnerable areas are illustrated in Figure 6.16. The outcome has been compared between the proposed algorithm and the CETESB result (Publicações e Relatórios | Águas Interiores, 2017). These areas have been characterized descriptively by colour codes, varying from deep red (very critical) to light red (less critical). The observation shows that most of the predicted critical zones are located close to the South East of SP with the dense and growing population, and rapid developments. As a result, WQ stresses are expected in these areas as

mapped by CETESB (i.e. 2016 results) as shown in Figure 6.17 for IQA, IVA and IET, respectively.

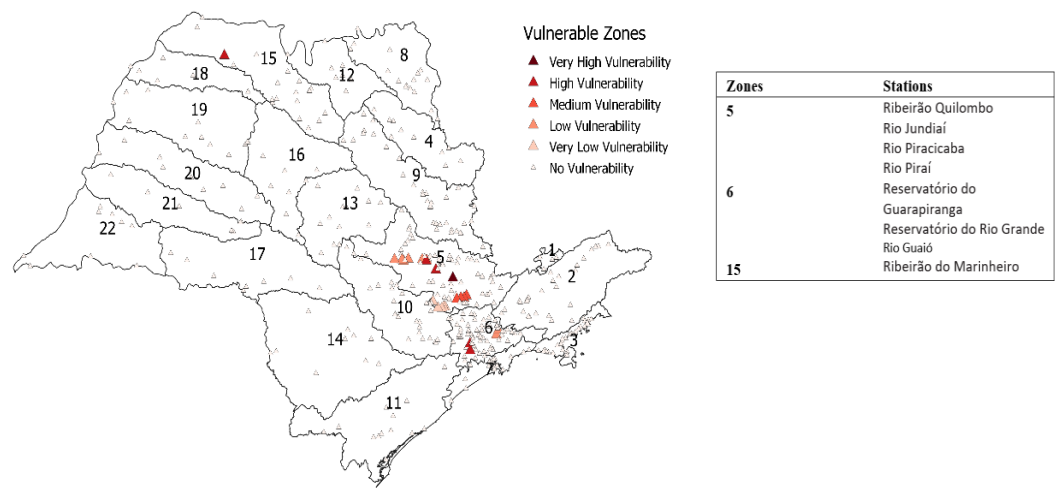


Figure 6. 16: Vulnerable zones identified by the Parity-Q-Deep Q network based on IQA, IET and IVA

From Figure 6.16 and 6.17, it is observable that the vulnerable areas are predicted by the proposed algorithm based on IQA, IET and IVA. It is also noticeable that the proposed algorithm can identify all the zones which are the most vulnerable.

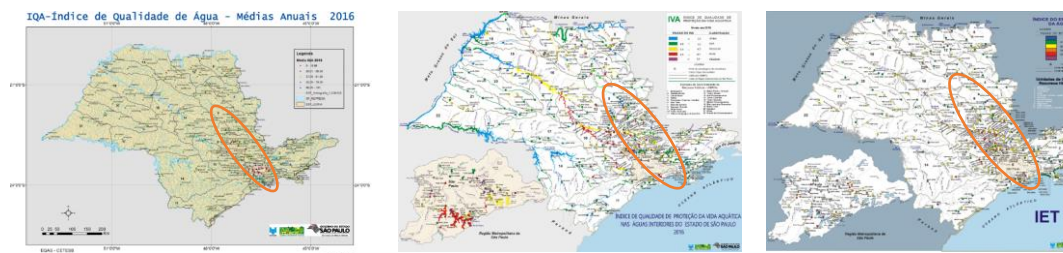


Figure 6. 17: a) CETESB (2016)’s IQA mapping; b) CETESB (2016)’s IVA mapping; (c) CETESB (2016)’s IET mapping.

From the interpretations of these comparisons and the observations in Figure 6.16 and 6.17, the predicted critical areas in this research reasonably conform to the observations of CETESB for WQI (CETESB, 2016). Therefore, this could be concluded that the results of the resilience prediction model are acceptable.

### 6.3.5 Statistical Evaluation of the proposed algorithm

The statistical test is done to explain the difference between the obtained results from each other on the basis of the variance between the sample means. This is common for the hypothesis testing that tells how significant these differences are. In this study, IGD values have been considered among the performance measuring criteria because of its capability to evaluate the convergence and diversity simultaneously. In addition, the smaller value of IGD stipulates all the generated solutions are on the true Pareto front.

The results are obtained by a two-tailed t-test with 38 degrees of freedom. The p-value and t-value are set at a 0.05 level of significance and  $\pm 2.02$  respectively. The result of Algorithm1  $\leftrightarrow$  Algorithm 2 is shown as “+” and “-” when Algorithm 1 is significantly better or worse compared to Algorithm 2, correspondingly. This process has been adopted from (Lwin, Qu and Kendall, 2014).

Table 6. 5: Student’s t-test results of different algorithms in different environments

Algorithm 1 Algorithm 2	Dynamic DST (Silver and Gold)	Dynamic DST (Attack by enemy)	Water Quality Resilience
PQDQN $\leftrightarrow$ MO-MCTS	+	-	+
PQDQN $\leftrightarrow$ MPDQN	+	+	+
PQDQN $\leftrightarrow$ MPQ	+	+	+
MO-MCTS $\leftrightarrow$ MPDQN	+	+	+
MO-MCTS $\leftrightarrow$ MPQ	-	+	+
MPDQN $\leftrightarrow$ MPQ	-	+	-

The result as shown in Table 6.5 demonstrates that the proposed algorithm outperforms other considered algorithms in almost every case except the dynamic DST (attack by enemy) instance where MO-MCTS performs better than the PQDQN in terms of the solution quality and run time.

In addition, MO-MCTS outperforms in each environment compared to the MPDQN in terms of solution quality. The result also shows that the MPDQN performs worse in every case except the dynamic DST (attack by enemy) where it



performs better than MPQ. MPQ performs worse in dynamic DST (silver and gold) environment compared to the MO-MCTS.

Therefore, this can be concluded that the proposed PQDQN has the best optimisation performance in general, considering all the dynamic environments compared to the other measured algorithms except for the dynamic DST (attack by enemy). In other words, the proposed algorithm has achieved the result with a good quality solution in terms of convergence and diversity.

#### **6.4 Strengths and Weaknesses of the Proposed Method**

##### **Strengths:**

- This MOMDP based PQDQN algorithm learns more properties and complex features by interacting within the environment using vector rewards. This approach reduces the feature-engineering part.
- This method can be fitted into unstructured data and implemented in the automatic network of CETESB dataset for real-time tracking to detect the critical zones based on the WQR. There is an overabundance of the unstructured data in various forms such as images, text, dialogue system, sensor data, and so forth. This utilisation may transfigure different domains, such as portfolio optimisation, healthcare, manufacturing, banking, stock exchange, aviation, e-commerce and so on.
- Another useful feature of this method is applying the preference value along with the parity value (i.e. dynamic value) allocation. It helps to set the users' preference to make the algorithm more robust and adjustable.

### Weaknesses:

- Formalising MOMDP in DRL settings, generally requires having a lot of parameters and a pre-defined model in terms of states, actions and rewards functions. For such applications, there is a need to gather a sufficiently large volume of data to train; otherwise, the model can suffer from overfitting. This portion of the ML study is typically a time-consuming activity (Leger et al., 2017; Schmidhuber, 2015).
- The learning process in the deep layer of the deep reinforcement learning paradigm is often hard to interpret. In this method, the agent naturally requires colossal computational power and time to be trained. The rationale of doing so is the divide and conquer attitude of each DQN that produce the  $\vec{Q}(s, a)$  to select the optimal policy.
- AI-enabled any decision support system is generally questionable to the new legislation of the data security such as compliance with the GDPR (EUGDPR – Information Portal, 2018) where any automated DSS should have legal bindings and a detailed explanation and consent from the user (Paramita Ghosh, 2018).

### 6.5 Limitations and Areas for Improvement

There is a famous aphorism in machine learning research, “*All models are wrong, but some models are useful*”- by George Box (Box and Draper, 1987). To some extent, this proverb is true because all the models are simplified versions of reality. Like every other solution, the proposed model also has the limitations and drawbacks such as it cannot solve all the dynamic problems. This can only solve a pre-defined model that uses MOMDP.

In the context of the developing dynamic multi-objective optimisation benchmark, the benchmark based on type 2, 3, and 4 has only been considered. There is an opportunity to extend this work for type 1 (i.e. optimal PS fluctuates where the optimal PF remains unmoved). Overall, the benchmarks can be further modified by introducing different obstacles which may change over time. In this study, up to three objectives are considered (i.e. DST attack by enemy). It is possible to increase the objectives and convert this benchmark into many objectives problem.

As the benchmark is created based on a grid-world gaming environment, the gameplay can be formulated in such a way that more obstacles and objectives can be integrated. Considering the proposed algorithm, only Q learning has been implemented and tested, the model-based approach also needs to be explored and investigated further for this scenario.

Considering the application of the proposed algorithm, water quality has been considered. However, the recent approach and implementations of DRL such as images as input were not considered. To improve the model, precipitation and the time-series of the whole dataset can be taken into consideration.

#### **Areas for improvement:**

- I. Forming the environment in a stochastic and non-stationary partially observable MOMDP.
- II. Comparing with different models of architectures, training parameters and reward functions.
- III. Improving the abilities of the method in sediment and automatic networks for the WQR dataset in São Paulo, Brazil.
- IV. Evaluate the method with real end-users.

- V. Incorporating parallel computing for faster computational performance.

## 6.6 Summary

Designing an appropriate dynamic environment is the key task in studying DMOPs using RL. Throughout this chapter, the evaluating criteria and performance have been evaluated for the proposed benchmark and the real-world scenario. In summary, the proposed DRL based model has utilised in the following categories:

- The benchmark has successfully satisfied the dynamic behaviour in the RL settings where there is more than one objective, possibly conflicting objectives.
- The proposed algorithm has successfully implemented in both test cases and outperforms the other algorithms except in the case of the dynamic DST (attack by enemy) environment.
- The findings in this chapter show an effective way to analyse the algorithms for DRL especially in the context of the DMOPs.
- A formulation of the water quality dataset has been presented as an MOMDP to identify the critical stations in the targeted area (i.e. São Paulo, Brazil) with the definitions of the system state, control action and reward function which are the rudiments in the proposed DRL based method.
- Furthermore, the network architecture has been developed to get the optimal policy in each DQNs for each episode generated by the emulator.

Throughout this chapter, the proposed algorithm is also critically evaluated from a neutral stand. As the PQDQN works based on the policy-search method, it is observable that the performance is affected by the direct representation of the policy mapping as discussed in the previous chapter. The proposed algorithm outperforms other considered algorithms which are observable with the performance measures (i.e. GD, IGD and HV) except for the DST (attack by enemy) environment where there is a random attack by the enemy submarine. This chapter also sheds light on the statistical tests (i.e. Student's T-test) to analyse the performance of the proposed algorithm. The variation of the performance in a stochastic domain can be resolved if the randomness can be eradicated in advance. The evaluation process has also enlightened the potential of using RL to solve various ML problems where there is a high-dimension or partially observable environment. In these scenarios, the agent may learn the behaviours from thousands even millions of elementary stages.

Based on the obtained results and corresponding analysis, the findings have been demonstrated critically where the algorithms performed with respect to the predefined experimental setups. It is worth mentioning that the attained results may vary in a different setting. Therefore, the comparisons have been presented with similar settings for every considered algorithm to avoid any bias. Then, for each algorithm, its average rank is considered.

Furthermore, the hyperparameters were selected cautiously so that the outcome may not lead to wrong assumptions or can draw an incorrect perception. Therefore, to test the algorithms, the best-known POFs are selected for both test cases. Still, further research is necessary to determine the best performance measure(s) for the cases where POF is unknown.

## Chapter 7

### Concluding Remarks and Future Directions

In this study, deep reinforcement learning method is investigated to address three challenges in the domain of DMOPs in RL settings. The challenges include establishing a benchmark and developing an algorithm for DMOPs, and its application in the real-world scenario. The absence of a benchmark in the area of DMOP in RL settings has successfully been identified. Later, a benchmark has been created based on the classic DST. The extended and modified version of the DST has fruitfully satisfied the dynamics in terms of changing parameters of a pre-defined environment. These variations are based on the changing optimal PF and PS over time.

In this thesis, a comprehensive analysis and a recent trend of DRL were discussed in the second chapter. The chapter has also highlighted the core elements and the necessary components for RL and DRL. Moreover, the test cases and the justification for doing this empirical research have been mentioned in the second chapter. The methodology has been discussed in the following Chapter 3. The proposed benchmark has been discussed in Chapter 4 where three environments have been created based on the dynamics that are presented by (Farina, Deb and Amato, 2004). These three environments are created based on the random treasure values, silver and gold treasure values, and the dynamic DST (attack by enemy) where a new objective has been integrated.

The problem definition and the mathematical model have also been discussed with the explanation of the network architecture and their corresponding hyperparameters in the fourth chapter. Two different experimental setups have been used for two different test cases that are common for every algorithm for fair comparison and analysis. Technically, in both cases, Keras framework is used with the backend of the TensorFlow based on the central processing unit (CPU) only.

Thus, this study has underlined an easy implementation of the MOMDP based system in the real-world scenario and solve an actual problem using the trial-error method. Hardcode approach of formalising the MOMDP is used to solve the problem instead of using the raw images as input. Chapter 5 sheds light on the proposed algorithm.

It has been observed that the proposed algorithm (i.e. PQDQN) enables the decomposition of problems into sub-problems and make a relationship among different objectives. This is nothing but the mapping between different conflicting objectives to get a compromising solution that adheres to the POF. It also provides a method for robust manipulation of priorities after the training which may ignore a specific behaviour or objective provided by a DQN. It permits a new objective to accommodate in the current settings because of the decompositions by different DQNs and combining with the preference values (e.g., if any).

The results in Chapter 6, perceives forming a Meta-policy that can have a significant impact on governing the optimal policy that needs to be selected in a particular state in an MOMDP. Besides, successful implementation of an MOMDP in a real-world scenario has been anticipated where the trained agent outperforms multi-policy algorithms such as MO-MCTS, MPQ, and MPDQN. Although the agent requires slightly higher elapsed time compared to the MO-MCTS agent in the dynamic DST (attack by enemy) environment, its accuracy in finding the Pareto optimum solutions is improved. Regarding the implementation, testing and validation of the proposed algorithm in a real-world situation, the dataset for water quality resilience in SP, Brazil has been used.

The proposed algorithm is based on the DRL that has successfully identified the most vulnerable zones which belong to the zone of 5 (*Ribeirão Quilombo*), 6 (*Reservatório do Guarapiranga*) and 15 (*Ribeirão do Marinho*). This experiment also shows how to use an MOMDP using DRL to solve a practical problem. The result also confirms that in a stochastic MOMDP setting where there

is randomness, the proposed algorithm performs better than MPQ and MPDQN.

In a nutshell, the contributions made in this thesis are as follows:

- It is the first study to create a new benchmark for DMOP in RL settings. This benchmark satisfies the dynamics of type II, III and IV.
- A novel algorithm (PQDQN) has been developed that can handle multi-objective using different DQNs based on DRL.
- Objective-relation mapping (ORM) is used for the first time to formulate meta-policy.
- An innovative method has been implemented to identify and predict the vulnerable zones based on water quality resilience in São Paulo, Brazil and
- A research gap has been identified through an extensive review of the existing state-of-the-art techniques and literature in the context of RL.

Hence, the concluding remarks are divided into the following sub-sections.

## **7.1 Final Remarks on the Benchmark**

To conclude the argument on the proposed benchmark, it is now established that there was a lack of a standard in the context of DMOPs study in RL domain. Therefore, it was important to investigate and analyse it further in the context of DMOPs in RL settings. Consequently, it was much needed to develop a benchmark that can handle the dynamics in a multi-objective environment. The benchmark was proposed in a way so that it does not deviate from the conventional settings (e.g., games, grid-world, and control problems) of the RL research.

Therefore, the benchmark was designed based on the existing testbed which is a modified version of the classical DST problem for RL research. Furthermore, the benchmark is designed to make it robust and scalable for the purpose of analysis



and evaluation. This may allow the researchers to modify, improve and compare different settings and analyse the benchmark further.

The proposed benchmark for DMOPs can satisfy the types that have been mentioned in the problem definition section which are type II, III and IV based on the changing optimal PF and PS. From the proposed dynamic environments, the purely random environment has been excluded where the treasure values change arbitrarily. The reason for excluding this environment was not to have any known patterns in this scenario and ML inconsistency in such circumstance. Likewise, the benchmark does not address the type I problem where optimal PS varies, and the optimal PF remains unchanged. Incorporating this scenario may have a significant impact on further investigation and may contribute to some new knowledge in the context of RL and DMOPs.

Another important observation is that the benchmark is not only capable of handling dynamics of the different RL settings but also provides the facilities to handle the changing objective such as in the attack by the enemy submarine DST environment. It is also possible to incorporate more than one enemy submarine or the existing enemy submarine that follows a pattern to hit the good agent and gives a tough time to the opponent to survive. This process surely will generate some new dynamics in this grid-world.

Therefore, the benchmark has got every potential to be enhanced further and become an appropriate testbed for solving dynamic optimisation problems in RL backgrounds. It is to be mentioned that this thesis does not take any credit of the original version of the DST testbed which was undoubtedly a remarkable contribution in the context of RL and optimisation research.

## 7.2 Final Remarks on Test Case 1

Test case 1 is related to the proposed benchmark. This test case is based on the dynamic environments that use the MOMDP model. Here, the vector rewards are utilised by the agent to find out the optimum solution over the scalar reward. PQDQN outperforms the other considered algorithms in finding the solution that is closed to the true PF. The elapsed time for training and obtaining the average rewards was commendable for the proposed algorithm in this test case. The proposed algorithm shows its capability to obtain all the Pareto frontiers in every environment. The elapsed time to be trained varies for every run that agent witnesses for every setting. However, the convergence time of the proposed algorithm was higher in the DST (attack by enemy) environment compared to MO-MCTS.

The algorithm also reveals that the obtained GD values in test case 1 are better compared to the others except for the DST (attack by enemy) environment based on hyperparameters as mentioned in Chapter 4. Like GD, PQDQN performs identically for HV concerning the evaluation of the performance measurements for the considered algorithms. The possible reason might be the enemy submarine's behaviour that is generated arbitrarily. Conversely, the proposed PQDQN outperforms other considered algorithms in terms of the IGD. The statistical test (i.e. Student's t-test) also signifies the same result that the PQDQN outperforms the other algorithms except for the dynamic DST (attack by the enemy) environment.

The dynamic weight allocation based on the Q values of each DQN has unlocked the opportunities to deal with the changing behaviour of the DMOPs and may work with the new objective even without re-training. This Q-function learned by the agent is represented in the tabular form which is one output for each tuple. In the PQDQN, the agent updates its target network to keep twiddling the policy as long as the performance gets better. Consequently, the converging time increases as space gets larger.

The suggested algorithm is based on the beliefs of the Q learning and it has two major pillars that distinguish from the single objective agent. This belongs to the vector reward and the learning capability of all the set of non-dominated solutions at the same time. However, in the light of the achieved results, it looks sensible to speculate that in the context of the dynamic environment, PQDQN tends to need less training steps compared to the other algorithms. This is due to the reward structure and the exploration mechanism of the proposed algorithm. It is to be argued that the Q learning approach may suffer from the overfitting and absolute greedy attitude to reach the goal.

With the help of the obtained results, it is clear that the proposed algorithm provides the way of storing the utility information which does not need a model either for learning or for selecting an action. As the Q learning algorithm is fundamentally based on the model-free approach, thus, the proposed algorithm achieves the obtained Pareto front by mapping multi-objective based on ORM as described in Chapter 5. Consequently, the agent is capable of reaching the supported solutions in a reasonable time frame. However, it could be argumentative that in the large space, a policy search-based method might not be the appropriate solutions as it often uses a stochastic policy. However, due to the policy selection mechanism mentioned in Chapter 5, it is noticeable that even in the dynamic search space, the proposed algorithm can perform and converge within the defined episodes.

Besides, the convolutional layers that are used in the network architecture by the proposed algorithm are two for test case 1. The agent also utilises Adam optimisers for this test case. These setups also emphasise the capability to fit in the real-world scenario compared to other deep learning algorithms such as AlexNet which has 5 convolutional layers (Krizhevsky, Sutskever and Hinton, 2012). Therefore, this study shows an easy formation of customising the layers in the DRL architecture where MOMDP is being used. The further implementation of different layers would also be appreciable and could be an impressive extension of the current setting.

It is worth mentioning that the proposed algorithm deals with the vector rewards in a 2-dimensional space. However, a fundamental and new knowledge would be the creation and investigation of the compatibility of the reward distribution in a three dimensional or an n-dimensional space. It should also be stated that a non-stationary policy may be unsuitable in a specific application as mentioned in (Vamplew et al., 2017a; Roijers et al., 2013). Therefore, the current algorithm still needs to be better for identifying the strategies for stationary deterministic policy. However, a recent approach of the Pareto-set algorithm as mentioned in (Moffaert and Nowé, 2014; Moffaert, Drugan and Nowé, 2014) represents an encouraging new tactic for identifying such policies.

The whole procedure of establishing the benchmark and using it as a test case was successful. Consequently, the proposed algorithm has implemented in the benchmark without any constraints and thus, the study accomplished its objectives 1, 2, 3 and 4 as mentioned in Chapter 1. This also answers the research question 1 which efficaciously fills the gap to establish the benchmark and address the problem settings by the proposed algorithm. All in all, this also signifies that test case 1 can be related to many problems in the domain of computer vision where the PF and PS changes or remains invariant.

### **7.3 Final Remarks on Test Case 2**

To accomplish the urgings on test case 2 which is related to the prediction of the vulnerable zones based on the water quality resilience, this thesis has successfully addressed this problem. This problem has addressed using one of the state-of-the-art machine learning techniques such as DRL. This also leads to conducting a feasibility study to investigate the potential and suitability of using RL methods for WQR predictions to identify and predict the critical zones in the domain of water engineering applications. Machine learning methods, particularly, artificial neural network, has been widely used in water systems engineering studies with fairly satisfactory performance as discussed in Chapter 2.

On the other hand, a new approach such as using RL and MOMDP to study water quality resilience may bring an effective solution in preparing the engineering systems. This approach also may help to tackle and cope with emerging challenges in the hydroinformatics domain. This study combines the ML and WQR which can provide an opportunity for more effective adoption of the resilience to face the various challenges and help to provide a sustainable environment. In this thesis, a dataset has been utilised which is produced by the CETESB from the state of São Paulo, Brazil. The agents' task was to identify and predict the critical zones where the IQA values are less and the IET and IVA values are high as described in Chapter 4 and 5. Thus, the last objective of this research work has been achieved.

Therefore, successful implementation of RL settings in predicting water quality resilience is a new paradigm and can enrich the implementation of AI and ML technologies to study the hydroinformatics and hydrodynamics arena. Its impact and formalising MOMDP in this context can be argumentative from an academic point of view. However, the researcher community cannot decline the future possibilities of using DRL in different industries including water treatment, smart cities, sustainable drainage systems, wearable computing and so on.

Moreover, this study can also have a significant impact on reactive and planned maintenance for water supply. It also shows a direction for the researchers to implement the proposed model in various water reservoirs to measure water quality resilience, which may help to reduce the efforts of manual data collection and support to build a sustainable environment.

To sum up, the implementation of DRL methods can be improved using various ML techniques and carried out further from this point. It can be concluded that test case 2 successfully answers the second research question and helps to achieve the final objective to reach the intended research goal. Besides, a new approach to solving an emerging problem such as water quality can always enlighten hidden potential associated with it.

## 7.4 Future Works

The AI community has observed a different dimension of solving the problem using RL. By combining the RL approach and traditional supervised machine learning, it is possible to augment machine learning with expert knowledge. As a result, embedding the deep neural network with RL can be fed with the expert input and the RL can provide the flexibility to use the knowledge in the new context respectively. Besides, RL agents are expected to be outperforming humans in many situations such as playing Go, Chess, Atari 2600 and so on (Li, 2017).

Despite the successes of DRL, a proper investigation needs to be made before applying it to solve a real-world and complex problem because of its overwhelming computational costs. A deep learning amalgamation with other traditional AI approaches can significantly improve the performance and achieve the desired goal. Possibly, it is not too far when AI systems will be able to acquire knowledge and act like a human in increasingly complicated environments (Enriquez et al., 2020).

In the context of the algorithm development, the off-policy algorithms have been considered predominantly. However, a model-based and on-policy algorithm can be explored further to investigate the applicability and efficacy in this context. The scope of the PQDQN can be enhanced in many objectives scenario. Besides, incorporating parallel computing can significantly reduce the elapsed time of being trained for the developed algorithm.

The proposed algorithm is being tested for the two objectives scenario (i.e. DST (silver and gold) and WQR model) and the three objectives (DST attack by the enemy) settings. At this stage, the immediate application will be using deep deterministic policy gradient (Lillicrap et al., 2015) to train the agent. After that, the ANOVA test will be performed to get an improved insight for statistical analysis. However, many objectives (i.e. more than four objectives) scenario is also in the plan to be implemented in the future (Jaimes and Coello, 2015).

To handle the partially observable, nondeterministic and uncertain environment, there are still many challenges that need to be considered such as benchmark evaluation in the dynamic environment as well as the performance measuring criteria in such benchmarks. Furthermore, the connections between AI and the real-world problems need to be more coupled so that practical problems can have effective as well as realistic solutions with an explainable format (e.g. XAI) to simplify the understanding and usage of AI (Barredo Arrieta et al., 2020). Besides, the findings from test case 2 with an automatic decision process need to be further investigated to comply with the existing systems.

Additionally, hierarchical reinforcement learning (HRL) can be used to extend the traditional (RL) methods to solve more complex tasks such as working with many objectives problems (Levy, Platt and Saenko, 2018). However, the majority of current HRL methods require on-policy training and these methods are difficult to apply in real-world scenarios. In a highly complex setting such as in a dynamic environment with the high-variance behaviours of the constraints, objectives or parameters, this approach can be useful.

However, this may be computationally expensive whereas the cost of a few million steps will be required to just do a simple step along with several days for training (Nachum et al., 2018). The ultimate vision of AI since its inception has been to construct autonomous agents that can interact with the environment and amongst each other. To achieve this level of success, multi-agent RL (MARL) may be a good solution that matches with the vision (Lopes Silva et al., 2019).

Nevertheless, the success of the MARL agent yet to be explored in the future (Kapoor, 2018). It is to be mentioned that, Vinyals et al., (2017) has introduced “The StarCraft II Learning Environment” which is a testbed for MARL research. This allows for granular control over the objectives and constraints in the pre-defined map of the environment. Incorporating these two approaches into the proposed algorithm can bring a hidden success for the DMOPs which may bring a

new horizon in the operational research (OR). Thus, RL can be suitable for the environment where the agent's task is to interact within the environment and win by self-learning. Therefore, multi-agent implementation can be utilised to solve a particular problem in the future which has not been considered in this study to identify the better policy. Additionally, human-agent teamwork can also be integrated to make a better and on-demand resolution in real-time for the decision-making process (van Wissen et al., 2012). Applying this technique in this benchmark may enhance the performance by allowing cooperation that can help to build a smarter decision-making scheme.

The parallel approach of solving DMOPs such as incorporating parallel computing can be scrutinised in the future to produce more competent and realistic solutions. The study of the dynamic applications will enhance more viable solutions for well-being and enhance the quality of life. In addition, the training and convergence time can be improved significantly in future with the help of using edge AI (Li et al., 2019) technique. This can also be helpful to implement in many practical applications such as for smart homes, Internet of Things (IoT) appliances (Chen and Giannakis, 2017; Liu, Zhang and Wang, 2018) and so on, considering our daily life is also dynamic with many constraints.

The DRL approach can also be operated to solve various metaheuristics problems (Torres-Jiménez and Pavón, 2014) such as in healthcare (Liu et al., 2017), chemical reaction optimisation (Bechikh, Chaabani and Ben Said, 2015), artificial immune systems (Azzouz, Bechikh and Said, 2012), radiation optimisation (Feng et al., 2018), particle swarm optimisation (Hein et al., 2016; Zhang et al., 2015), and ant colony optimisation (Desell et al., 2015).

On the other hand, in the near future, access to clean water will be more of a strategic issue. Besides, climate change will also have a major impact on drinking water quality and its availability. The manual process of data collection can be very time-consuming and thus, it will be replaced by the automatic process



eventually. As a result, integrating different machine learning methods can have a significant impact in this domain.

Therefore, the cutting-edge technologies such as DRL may bring even greater success to achieve human-level expertise to determine not only the vulnerable zones but also the associated reasons to be vulnerable such as the influence of drought or precipitation. In terms of integrating real-world scenarios, raw pixels of the environment can be embedded for an interesting enhancement of this work which may help to study in computational fluid dynamics, ocean engineering, naval hydrodynamics, a study of marine and aquatic life, and water conservancy. This can also be implicated in the conventional study of ML such as in video games, stock-exchange prediction, image processing and so on.

Lastly, as an ending note, the readers are reminded of Isaac Asimov's (Asimov, 1950) law of the agents' saying that we must create the agent which is helpful to human. This can be achieved by applying governance and ensuring the facilities of tractability (Matt O'Brien, 2020). In this thesis, the discussion on the possibility of vulnerable or dangerous AI has been eschewed. The author would like to leave the discussion to an open-end so that the readers can have their own portrayal of future AI, be it a positive or negative one. However, the author believes that mankind will be more benefitted from AI and the self-learning ability of the machines if they can imagine the revolutionary outcomes of it. Finally, the author would like to conclude the thesis with a quote of Alan Turing, the famous British scientist and the computer science pioneer:

*"Those who can imagine anything, can create the impossible."*

— Alan Turing

## **Ethical Considerations**

This study proves the concept of optimisation in the dynamic multi-objective environment. To satisfy this, computer-generated simulated environments have been considered. In other words, gaming environments have been incorporated to do the experiments. Therefore, human participation was not required during the research period.

## References

- Aberdeen, D., Thiébaux, S. and Zhang, L., 2004. Decision-Theoretic Military Operations Planning. [online] Available at: <[www.aaai.org](http://www.aaai.org)> [Accessed 8 Sep. 2018].
- Aimin Zhou, Yaochu Jin and Qingfu Zhang, 2014. A Population Prediction Strategy for Evolutionary Dynamic Multiobjective Optimization. *IEEE Transactions on Cybernetics*, [online] 44(1), pp.40–53. Available at: <<http://ieeexplore.ieee.org/document/6471286/>> [Accessed 10 Oct. 2018].
- Algoul, S., Alam, M.S., Hossain, M.A. and Majumder, M.A.A., 2011. Multi-objective optimal chemotherapy control model for cancer treatment. *Medical & Biological Engineering & Computing*, [online] 49(1), pp.51–65. Available at: <<http://link.springer.com/10.1007/s11517-010-0678-y>>.
- Amato, P. and Farina, M., 2005. An ALife-Inspired Evolutionary Algorithm for Dynamic Multiobjective Optimization Problems. In: *Soft Computing: Methodologies and Applications*. [online] Berlin/Heidelberg: Springer-Verlag. pp.113–125. Available at: <[http://link.springer.com/10.1007/3-540-32400-3\\_9](http://link.springer.com/10.1007/3-540-32400-3_9)> [Accessed 9 Sep. 2018].
- Amitrano, D., Martino, G. Di, Iodice, A., Mitidieri, F., Papa, M.N., Riccio, D. and Ruello, G., 2014. Sentinel-1 for Monitoring Reservoirs: A Performance Analysis. *Remote Sensing*, [online] 6(11), pp.10676–10693. Available at: <<http://www.mdpi.com/2072-4292/6/11/10676>> [Accessed 8 Sep. 2018].
- Andersen, A., Goodwin, M. and Granmo, O.-C., 2018. The Dreaming Variational Autoencoder for Reinforcement Learning Environments. [online] Available at: <<https://arxiv.org/pdf/1810.01112.pdf>> [Accessed 17 Nov. 2018].
- Anschel, O., Baram, N. and Shimkin, N., 2017. Averaged-DQN: Variance Reduction and Stabilization for Deep Reinforcement Learning. [online] Available at: <<http://proceedings.mlr.press/v70/anschel17a/anschel17a.pdf>> [Accessed 9 Sep. 2018].
- Antanasijević, D., Pocajt, V., Perić-Grujić, A. and Ristić, M., 2014. Modelling of dissolved oxygen in the Danube River using artificial neural networks and Monte Carlo Simulation uncertainty analysis. *Journal of Hydrology*, [online] 519, pp.1895–1907. Available at: <<http://linkinghub.elsevier.com/retrieve/pii/S0022169414007896>> [Accessed 9 Sep. 2018].
- Apshvalka, D., Donina, D. and Kirikova, M., 2009. Understanding the Problems of Requirements Elicitation Process: A Human Perspective. [online] *Information Systems Development*. Springer. Available at: <<https://pdfs.semanticscholar.org/3610/338d653c1e3c71e3ec1d5302126ae7f39d12.pdf>> [Accessed 8 Dec. 2018].
- Artificial intelligence: The saviour of mankind or the end of the world?. 2018. [online] *Telegraph*. Available at: <<https://www.telegraph.co.uk/technology/2018/02/21/artificial-intelligence-saviour-mankind-end-world/>> [Accessed 27 Oct. 2018].
- Arulkumaran, K., Deisenroth, M.P., Brundage, M. and Bharath, A.A., 2017. Deep Reinforcement Learning: A Brief Survey. *IEEE Signal Processing Magazine*, [online]

34(6), pp.26–38. Available at: <<http://ieeexplore.ieee.org/document/8103164/>> [Accessed 8 Sep. 2018].

Asimov, I., 1990. Robot visions. Penguin Books.

Azzouz, R., Bechikh, S. and Ben Said, L., 2014. A Multiple Reference Point-based evolutionary algorithm for dynamic multi-objective optimization with undetectable changes. In: 2014 IEEE Congress on Evolutionary Computation (CEC). [online] IEEE, pp.3168–3175. Available at: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6900569>> [Accessed 9 Sep. 2018].

Azzouz, R., Bechikh, S. and Ben Said, L., 2015. Multi-objective Optimization with Dynamic Constraints and Objectives. In: Proceedings of the 2015 on Genetic and Evolutionary Computation Conference - GECCO '15. [online] New York, New York, USA: ACM Press, pp.615–622. Available at: <<http://dl.acm.org/citation.cfm?doid=2739480.2754708>> [Accessed 9 Sep. 2018].

Azzouz, R., Bechikh, S. and Ben Said, L., 2017. A dynamic multi-objective evolutionary algorithm using a change severity-based adaptive population management strategy. *Soft Computing*, [online] 21(4), pp.885–906. Available at: <<http://link.springer.com/10.1007/s00500-015-1820-4>> [Accessed 9 Sep. 2018].

Azzouz, R., Bechikh, S. and Ben Said, L., 2017. Dynamic Multi-objective Optimization Using Evolutionary Algorithms: A Survey. [online] pp.31–70. Available at: <[http://link.springer.com/10.1007/978-3-319-42978-6\\_2](http://link.springer.com/10.1007/978-3-319-42978-6_2)> [Accessed 9 Sep. 2018].

Azzouz, R., Bechikh, S. and Ben Said, L., 2012. Articulating Decision Maker's Preference Information within Multiobjective Artificial Immune Systems. In: 2012 IEEE 24th International Conference on Tools with Artificial Intelligence. [online] IEEE, pp.327–334. Available at: <<http://ieeexplore.ieee.org/document/6495064/>> [Accessed 11 Nov. 2018].

B. Bhattacharya, A.H. Lobbrecht and D.P. Solomatine, 2002. Control of water levels of regional water systems using reinforcement learning. In: 5th International Conference on Hydroinformatics, Cardiff, UK. [online] Cardiff, UK. Available at: <[https://www.researchgate.net/publication/237139834\\_Control\\_of\\_water\\_levels\\_of\\_regional\\_water\\_systems\\_using\\_reinforcement\\_learning](https://www.researchgate.net/publication/237139834_Control_of_water_levels_of_regional_water_systems_using_reinforcement_learning)> [Accessed 8 Sep. 2018].

Badar, A., Umre, B.S. and Junghare, A.S., 2014. Study of Artificial Intelligence Optimization Techniques applied to Active Power Loss Minimization. [online] Available at: <[www.iosrjournals.org](http://www.iosrjournals.org)> [Accessed 8 Dec. 2018].

Bahri, O., Ben Amor, N. and El-Ghazali, T., 2014. New Pareto Approach for Ranking Triangular Fuzzy Numbers. [online] Springer, Cham, pp.264–273. Available at: <[http://link.springer.com/10.1007/978-3-319-08855-6\\_27](http://link.springer.com/10.1007/978-3-319-08855-6_27)> [Accessed 19 Nov. 2018].

Baldi, P. and Sadowski, P., 2018. Learning in the machine: Recirculation is random backpropagation. *Neural Networks*, [online] 108, pp.479–494. Available at: <<https://www.sciencedirect.com/science/article/pii/S0893608018302685>> [Accessed 20 Nov. 2018].

- Bamakan, S.M.H., Nurgaliev, I. and Qu, Q., 2019. Opinion leader detection: A methodological review. *Expert Systems with Applications*, [online] 115, pp.200–222. Available at: <<https://www.sciencedirect.com/science/article/pii/S0957417418304950>> [Accessed 19 Nov. 2018].
- Barbosa, M.C., Alam, K. and Mushtaq, S., 2016. Water policy implementation in the state of São Paulo, Brazil: Key challenges and opportunities. *Environmental Science & Policy*, [online] 60, pp.11–18. Available at: <<https://www.sciencedirect.com/science/article/pii/S1462901116300405>> [Accessed 8 Sep. 2018].
- Barracough, P.A., Hossain, M.A., Tahir, M.A., Sexton, G. and Aslam, N., 2013. Intelligent phishing detection and protection scheme for online transactions. *Expert Systems with Applications*, [online] 40(11), pp.4697–4706. Available at: <<https://www.sciencedirect.com/science/article/pii/S0957417413001255>> [Accessed 8 Dec. 2018].
- Barredo Arrieta, A., Díaz-Rodríguez, N., Del Ser, J., Benetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R. and Herrera, F., 2020. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, [online] 58, pp.82–115. Available at: <<https://www.sciencedirect.com/science/article/pii/S1566253519308103>> [Accessed 5 Mar. 2020].
- Barrett, L. and Narayanan, S., 2008. Learning all optimal policies with multiple criteria. In: *Proceedings of the 25th international conference on Machine learning - ICML '08*. [online] New York, New York, USA: ACM Press. pp.41–47. Available at: <<http://portal.acm.org/citation.cfm?doid=1390156.1390162>> [Accessed 9 Sep. 2018].
- Başçö, E. and Orhan, M., 2000. Reinforcement Learning and Dynamic Optimization Reinforcement Learning and Dynamic Optimization. [online] *Journal of Economic and Social Research*, Available at: <<https://www.researchgate.net/publication/4728155>> [Accessed 17 Nov. 2018].
- Becher, V., Carton, O. and Heiber, P.A., 2015. Normality and automata. *Journal of Computer and System Sciences*, [online] 81, pp.1592–1613. Available at: <[www.elsevier.com/locate/jcss](http://www.elsevier.com/locate/jcss)> [Accessed 21 Nov. 2018].
- Bechikh, S., Chaabani, A. and Ben Said, L., 2015. An Efficient Chemical Reaction Optimization Algorithm for Multiobjective Optimization. *IEEE Transactions on Cybernetics*, [online] 45(10), pp.2051–2064. Available at: <<http://www.ncbi.nlm.nih.gov/pubmed/25373137>> [Accessed 11 Nov. 2018].
- Bechikh, S., Datta, R. and Gupta, A.K., 2018. Recent advances in evolutionary multi-objective optimization.
- Bellemare, M.G., Naddaf, Y., Veness, J. and Bowling, M., 2012. The Arcade Learning Environment: An Evaluation Platform for General Agents. *Journal of Artificial Intelligence Research*. [online] Available at: <<http://arxiv.org/abs/1207.4708>> [Accessed 4 Nov. 2018].
- Bellman, R., 1958. Dynamic programming and stochastic control processes. *Information and Control*, [online] 1(3), pp.228–239. Available at:

<<https://www.sciencedirect.com/science/article/pii/S0019995858800030>> [Accessed 8 Sep. 2018].

Benedikt Frey, C., Osborne, M.A., Dewey, D., Dorn, D., Flint, A., Goldin, C., Muellbauer, J., Mueller, V., Newman, P., Ó hÉigeartaigh, S., Sandberg, A., Shanahan, M. and Woolcock for their excellent suggestions, K., 2013. The future of employment: how susceptible are jobs to computerisation? \*. [online] Available at: <[https://www.oxfordmartin.ox.ac.uk/downloads/academic/The\\_Future\\_of\\_Employment.pdf](https://www.oxfordmartin.ox.ac.uk/downloads/academic/The_Future_of_Employment.pdf)> [Accessed 13 Oct. 2018].

Bertsekas, D.P. and Tsitsiklis, J.N., 1996. Neuro-dynamic programming. [online] Athena Scientific. Available at: <<http://athenasc.com/ndpbook.html>> [Accessed 8 Sep. 2018].

Binois, M., Rulli re, D., Roustant, O. and ere, D., 2015. On the estimation of Pareto fronts from the point of view of copula theory. *Information Sciences*, [online] 324, pp.270–285. Available at: <<http://www.sciencedirect.com/science/article/pii/S0020025515004697>> [Accessed 17 Nov. 2018].

Bolisetti, S.K., Patwary, M., Soliman, A.-H. and Abdel-Maguid, M., 2017. RF Sensing Based Target Detector for Smart Sensing Within Internet of Things in Harsh Sensing Environments. *IEEE Access*, [online] 5, pp.13346–13363. Available at: <<http://ieeexplore.ieee.org/document/7986962/>> [Accessed 8 Dec. 2018].

Botte, M. and Sch bel, A., 2019. Dominance for multi-objective robust optimization concepts. *European Journal of Operational Research*, [online] 273(2), pp.430–440. Available at: <<https://linkinghub.elsevier.com/retrieve/pii/S037722171830715X>> [Accessed 19 Nov. 2018].

Box, G.E.P. and Draper, N.R., 1987. Empirical model-building and response surfaces. Wiley.

Boyan, J.A. and Moore, A.W., 1995. Generalization in Reinforcement Learning: Safely Approximating the Value Function. [online] MIT Press. Available at: <[https://www.ri.cmu.edu/pub\\_files/pub1/boyan\\_justin\\_1995\\_1/boyan\\_justin\\_1995\\_1.pdf](https://www.ri.cmu.edu/pub_files/pub1/boyan_justin_1995_1/boyan_justin_1995_1.pdf)> [Accessed 6 Dec. 2018].

Briot, J.-P., 2018. Deep Learning Techniques for Music Generation. SPRINGER. Available at:< <https://www.springer.com/gb/book/9783319701622>> [Accessed 11 Dec. 2018]

Brunner, R.J. and Kim, E.J., 2016. Teaching Data Science. *Procedia Computer Science*, [online] 80, pp.1947–1956. Available at: <<https://www.sciencedirect.com/science/article/pii/S1877050916310006>> [Accessed 4 Dec. 2018].

C mara, M., Ortega, J. and de Toro, F., 2009. Performance Measures for Dynamic Multi-Objective Optimization. [online] Springer, Berlin, Heidelberg.pp.760–767. Available at: <[http://link.springer.com/10.1007/978-3-642-02478-8\\_95](http://link.springer.com/10.1007/978-3-642-02478-8_95)> [Accessed 17 Nov. 2018].

- Camara, M., Ortega, J. and Toro, F.J., 2007. Parallel Processing for Multi-objective Optimization in Dynamic Environments. In: 2007 IEEE International Parallel and Distributed Processing Symposium. [online] IEEE.pp.1–8. Available at: <<http://ieeexplore.ieee.org/document/4228161/>> [Accessed 10 Oct. 2018].
- Carmen del Solar Valdés, 2017. Do AI applications need a persona? [online] Available at: <<https://www.artificial-solutions.com/blog/do-ai-applications-need-a-persona>> [Accessed 4 Dec. 2018].
- CETESB, Brazil. Filtration & Separation. 2005. [online] 37(5), p.39. Available at: <<https://www.sciencedirect.com/science/article/pii/S001518820088903X?via%3Dihub>> [Accessed 8 Sep. 2018].
- Chatzilygeroudis, K., Rama, R., Kaushik, R., Goepp, D., Vassiliades, V. and Mouret, J.-B., 2017. Black-Box Data-efficient Policy Search for Robotics. [online] Available at: <<https://hal.inria.fr/hal-01576683>> [Accessed 19 Jul. 2018].
- Chen, H., Li, M. and Chen, X., 2009. Using Diversity as an Additional-objective in Dynamic Multi-objective Optimization Algorithms. In: 2009 Second International Symposium on Electronic Commerce and Security. [online] IEEE.pp.484–487. Available at: <<http://ieeexplore.ieee.org/document/5209708/>> [Accessed 10 Oct. 2018].
- Chen, T. and Giannakis, G.B., 2017. Bandit Convex Optimization for Scalable and Dynamic IoT Management. [online] Available at: <<https://arxiv.org/pdf/1707.09060.pdf>> [Accessed 11 Nov. 2018].
- Chen, W.-B. and Liu, W.-C., 2015. Water Quality Modeling in Reservoirs Using Multivariate Linear Regression and Two Neural Network Models. *Advances in Artificial Neural Systems*, [online] 2015, pp.1–12. Available at: <<https://www.hindawi.com/archive/2015/521721/>> [Accessed 5 Dec. 2018].
- Cheng, R., Jin, Y., Narukawa, K. and Sendhoff, B., 2015. A Multiobjective Evolutionary Algorithm Using Gaussian Process-Based Inverse Modeling. *IEEE Transactions on Evolutionary Computation*, [online] 19(6), pp.838–856. Available at: <<http://ieeexplore.ieee.org/document/7018980/>> [Accessed 8 Sep. 2018].
- Cheng, R., Jin, Y., Olhofer, M. and sendhoff, B., 2017. Test Problems for Large-Scale Multiobjective and Many-Objective Optimization. *IEEE Transactions on Cybernetics*, [online] 47(12), pp.4108–4121. Available at: <<http://ieeexplore.ieee.org/document/7553457/>> [Accessed 8 Sep. 2018].
- Cheng, S., Shi, Y. and Qin, Q., 2012. On the Performance Metrics of Multiobjective Optimization. [online] Springer, Berlin, Heidelberg.pp.504–512. Available at: <[http://link.springer.com/10.1007/978-3-642-30976-2\\_61](http://link.springer.com/10.1007/978-3-642-30976-2_61)> [Accessed 17 Nov. 2018].
- Chi-Keong Goh and Kay Chen Tan, 2009. A Competitive-Cooperative Coevolutionary Paradigm for Dynamic Multiobjective Optimization. *IEEE Transactions on Evolutionary Computation*, [online] 13(1), pp.103–127. Available at: <<http://ieeexplore.ieee.org/document/4553723/>> [Accessed 10 Oct. 2018].
- Chou, J.-S., Ho, C.-C. and Hoang, H.-S., 2018. Determining quality of water in reservoir using machine learning. *Ecological Informatics*, [online] 44, pp.57–75. Available at:

<<https://www.sciencedirect.com/science/article/pii/S157495411730208X#f0005>>  
[Accessed 17 Apr. 2018].

Chris Graham, 2018. Artificial intelligence: The saviour of mankind or the end of the world? [online] Available at:  
<<https://www.telegraph.co.uk/technology/2018/02/21/artificial-intelligence-saviour-mankind-end-world/>> [Accessed 27 Oct. 2018].

Clarke, R., 2019. Principles and business processes for responsible AI. *Computer Law and Security Review*, 35(4), pp.410–422.

Collins, A.G.E. and Frank, M.J., 2012. How much of reinforcement learning is working memory, not reinforcement learning? A behavioral, computational, and neurogenetic analysis. *The European journal of neuroscience*, [online] 35(7), pp.1024–35. Available at: <<http://www.ncbi.nlm.nih.gov/pubmed/22487033>> [Accessed 17 Dec. 2018].

CONAMA Resolution 357/05 – Brazilian NR. 2005. [online] Available at:  
<<http://www.braziliannr.com/brazilian-environmental-legislation/conama-resolution-35705/>> [Accessed 5 Dec. 2018].

Cormen, T.H., 2009. *Introduction to algorithms*. MIT Press.

Crites, R., Crites, R. and Barto, A., 1996. Improving Elevator Performance Using Reinforcement Learning. *Advances in Neural Information Processing Systems* 8, [online] 8, pp.1017–1023. Available at:  
<<http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.17.5519>> [Accessed 8 Sep. 2019].

Cruz, C., González, J.R. and Pelta, D.A., 2011. Optimization in dynamic environments: a survey on problems, methods and measures. [online] Available at:  
<[http://www.aifb.uni-karlsruhe.de/\\*jbr/EvoDOP.](http://www.aifb.uni-karlsruhe.de/*jbr/EvoDOP.)> [Accessed 4 Nov. 2018].

Dan Clark, 2018. Top 16 Open Source Deep Learning Libraries and Platforms. [online] KDnuggets. Available at: <<https://www.kdnuggets.com/2018/04/top-16-open-source-deep-learning-libraries.html>> [Accessed 4 Dec. 2018].

Daniel Faggella, 2018. Machine Learning in Finance - Present and Future Applications - Artificial Intelligence Companies, Insights, Research. [online] Available at:  
<<https://emerj.com/ai-sector-overviews/machine-learning-in-finance/>> [Accessed 4 Dec. 2018].

Das, S., Dey, A. and Roy, N., 2015. Applications of Artificial Intelligence in Machine Learning: Review and Prospect. [online] *International Journal of Computer Applications*, Available at:  
<<https://pdfs.semanticscholar.org/6850/8ffc9f75462fd31de620d03093b214734011.pdf>> [Accessed 8 Dec. 2018].

Deb, K., 2011. Single and Multi-Objective Dynamic Optimization: Two Tales from an Evolutionary Perspective. [online] Available at: <<http://www.iitk.ac.in/kangal/deb.htm>> [Accessed 10 Oct. 2018].

Deb, K. and Jain, H., 2014. An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I: Solving



- Problems With Box Constraints. *IEEE Transactions on Evolutionary Computation*, [online] 18(4), pp.577–601. Available at: <<http://ieeexplore.ieee.org/document/6600851/>> [Accessed 8 Sep. 2018].
- Deb, K., Rao N., U.B. and Karthik, S., 2007. Dynamic Multi-objective Optimization and Decision-Making Using Modified NSGA-II: A Case Study on Hydro-thermal Power Scheduling. In: *Evolutionary Multi-Criterion Optimization*. [online] Berlin, Heidelberg: Springer Berlin Heidelberg. pp.803–817. Available at: <[http://link.springer.com/10.1007/978-3-540-70928-2\\_60](http://link.springer.com/10.1007/978-3-540-70928-2_60)> [Accessed 9 Sep. 2018].
- DeepMind, 2016. DeepMind AI Reduces Google Data Centre Cooling Bill by 40% | DeepMind. [online] Available at: <<https://deepmind.com/blog/deepmind-ai-reduces-google-data-centre-cooling-bill-40/>> [Accessed 8 Sep. 2018].
- Desell, T., Clachar, S., Higgins, J. and Wild, B., 2015. Evolving Deep Recurrent Neural Networks Using Ant Colony Optimization. [online] Springer, Cham. pp.86–98. Available at: <[http://link.springer.com/10.1007/978-3-319-16468-7\\_8](http://link.springer.com/10.1007/978-3-319-16468-7_8)> [Accessed 11 Nov. 2018].
- Deshpande, S., Watson, L.T. and Canfield, R.A., 2013. Pareto Front Approximation Using a Hybrid Approach. *Procedia Computer Science*, [online] 18, pp.521–530. Available at: <<https://www.sciencedirect.com/science/article/pii/S1877050913003591>> [Accessed 19 Nov. 2018].
- Drugan, M., Wiering, M., Vamplew, P. and Chetty, M., 2017. Special issue on multi-objective reinforcement learning. *Neurocomputing*.
- Duan, Y., Edwards, J.S. and Dwivedi, Y.K., 2019. Artificial intelligence for decision making in the era of Big Data – evolution, challenges and research agenda. *International Journal of Information Management*, 48, pp.63–71.
- Dwivedi, Y.K., Hughes, L., Ismagilova, E., Aarts, G., Coombs, C., Crick, T., Duan, Y., Dwivedi, R., Edwards, J., Eirug, A., Galanos, V., Ilavarasan, P.V., Janssen, M., Jones, P., Kar, A.K., Kizgin, H., Kronemann, B., Lal, B., Lucini, B., Medaglia, R., Le Meunier-FitzHugh, K., Le Meunier-FitzHugh, L.C., Misra, S., Mogaji, E., Sharma, S.K., Singh, J.B., Raghavan, V., Raman, R., Rana, N.P., Samothrakakis, S., Spencer, J., Tamilmani, K., Tubadji, A., Walton, P. and Williams, M.D., 2019. Artificial Intelligence (AI): Multidisciplinary perspectives on emerging challenges, opportunities, and agenda for research, practice and policy. *International Journal of Information Management*, p.101994.
- Einax, J.W., Aulinger, A., v. Tümpling, W. and Prange, A., 1999. Quantitative description of element concentrations in longitudinal river profiles by multiway PLS models. *Fresenius' Journal of Analytical Chemistry*, [online] 363(7), pp.655–661. Available at: <<http://link.springer.com/10.1007/s002160051267>> [Accessed 5 Dec. 2018].
- Elsafi, S.H., 2014. Artificial Neural Networks (ANNs) for flood forecasting at Dongola Station in the River Nile, Sudan. *Alexandria Engineering Journal*, [online] 53(3), pp.655–662. Available at: <<https://www.sciencedirect.com/science/article/pii/S1110016814000660>> [Accessed 5 Dec. 2018].

- Elshemy, M. and Meon, G., 2016. Water Quality Assessment of Aswan High Dam Reservoir. [online] pp.105–143. Available at: <[http://link.springer.com/10.1007/698\\_2016\\_96](http://link.springer.com/10.1007/698_2016_96)> [Accessed 5 Dec. 2018].
- Endel, F. and Piringer, H., 2015. Data Wrangling: Making data useful again. IFAC-PapersOnLine, [online] 48(1), pp.111–112. Available at: <<https://www.sciencedirect.com/science/article/pii/S2405896315001986>> [Accessed 14 Nov. 2018].
- Enriquez, J.G., Jimenez-Ramirez, A., Dominguez-Mayo, F.J. and Garcia-Garcia, J.A., 2020. Robotic Process Automation: A Scientific and Industrial Systematic Mapping Study. IEEE Access, 8, pp.39113–39129.
- Etchepare, R. and van der Hoek, J.P., 2015. Health risk assessment of organic micropollutants in greywater for potable reuse. Water Research, [online] 72, pp.186–198. Available at: <<http://www.ncbi.nlm.nih.gov/pubmed/25472689>> [Accessed 5 Dec. 2018].
- EUGDPR – Information Portal. 2018. [online]. Available at: <<https://eugdpr.org/>> [Accessed 11 Nov. 2018].
- Fakhreddine and Clarence, 2004. Soft Computing and Intelligent Systems Design: Theory, Tools and Applications by Fakhreddine O. Karray. [online] Available at: <<https://www.amazon.co.uk/Soft-Computing-Intelligent-Systems-Design/dp/B01K2OD438>> [Accessed 12 Dec. 2018].
- Farina, M., Deb, K. and Amato, P., 2004. Dynamic Multiobjective Optimization Problems: Test Cases, Approximations, and Applications. Transactions on Evolutionary Computation, [online] 8(5), pp.425–442. Available at: <<http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1347158>>.
- Federico Mor, 2018. What can artificial intelligence reveal about Brexit? – House of Commons Library. [online] Available at: <<https://commonslibrary.parliament.uk/science/technology/what-can-artificial-intelligence-reveal-about-brexit/>> [Accessed 19 Nov. 2018].
- Feinberg, E.A. and Shwartz, A., 2014. Handbook of Markov Decision Processes - Methods and Applications. p.93.
- Felix Yu, 2017. Deep Q Network vs Policy Gradients - An Experiment on VizDoom with Keras | Felix Yu. [online] Available at: <<https://flyyufelix.github.io/2017/10/12/dqn-vs-pg.html>> [Accessed 7 Dec. 2018].
- Feng, M., Valdes, G., Dixit, N. and Solberg, T.D., 2018. Machine Learning in Radiation Oncology: Opportunities, Requirements, and Needs. Frontiers in oncology, [online] 8, p.110. Available at: <<http://www.ncbi.nlm.nih.gov/pubmed/29719815>> [Accessed 11 Nov. 2018].
- Fernandez, P., McCarthy, I.P. and Rakotobe-Joel, T., 2001. An evolutionary approach to benchmarking. [online] An International Journal, # MCB University Press. Available at: <<http://www.emerald-library.com/ft>> [Accessed 5 Dec. 2018].

FN, C. and MF, M., 2017. Factors Affecting Water Pollution: A Review. *Journal of Ecosystem & Ecography*, [online] 07(01), pp.1–3. Available at: <<https://www.omicsonline.org/open-access/factors-affecting-water-pollution-a-review-2157-7625-1000225.php?aid=87940>> [Accessed 5 Dec. 2018].

French, J., Mawdsley, R., Fujiyama, T. and Achuthan, K., 2017. Combining machine learning with computational hydrodynamics for prediction of tidal surge inundation at estuarine ports. *Procedia IUTAM*, 25, pp.28–35.

Gábor, Kalmár and Szepesvári, C., 1998. Multi-criteria Reinforcement Learning. *Proceedings of the Fifteenth International Conference on Machine Learning*, [online] pp.24–27. Available at: <<https://dl.acm.org/citation.cfm?id=657298>> [Accessed 8 Sep. 2018].

Gambardella, L.M. and Dorigo, M., 1995. Ant-Q: A Reinforcement Learning approach to the traveling salesman problem. [online] Morgan Kaufmann. Available at: <<http://www.idsia.ch/http/iridia.ulb.ac.be/dorigo/dorigo.html>> [Accessed 4 Dec. 2018].

Gary Orenstein, 2018. Dawn of intelligent applications | InfoWorld. [online] Available at: <<https://www.infoworld.com/article/3247869/machine-learning/dawn-of-intelligent-applications.html>> [Accessed 4 Dec. 2018].

Gazzaz, N.M., Yusoff, M.K., Aris, A.Z., Juahir, H. and Ramli, M.F., 2012. Artificial neural network modeling of the water quality index for Kinta River (Malaysia) using water quality variables as predictors. *Marine Pollution Bulletin*, [online] 64(11), pp.2409–2420. Available at: <<https://www.sciencedirect.com/science/article/pii/S0025326X12004043>> [Accessed 5 Dec. 2018].

Gil Press, 2016. Cleaning Big Data: Most Time-Consuming, Least Enjoyable Data Science Task, Survey Says. [online] Available at: <<https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/#974189c6f637>> [Accessed 14 Nov. 2018].

Gil Press, 2018. When Machine Learning Started To Sense The World. [online] Available at: <<https://www.forbes.com/sites/gilpress/2018/01/07/when-machine-learning-started-to-sense-the-world/#3df4f39551f0>> [Accessed 8 Dec. 2018].

GitHub - RL-LDV-TUM/morlbench: Multiobjective Reinforcement Learning Benchmark Suite. 2018. [online] Available at: <<https://github.com/RL-LDV-TUM/morlbench>> [Accessed 17 Nov. 2018].

GitHub - ttajmayer/morl-dv: Modular Multi-Objective Reinforcement Learning with Decision Values. 2018. [online] Available at: <<https://github.com/ttajmayer/morl-dv>> [Accessed 17 Nov. 2018].

Glavic, M., Fonteneau, R. and Ernst, D., 2017. Reinforcement Learning for Electric Power System Decision and Control: Past Considerations and Perspectives. *IFAC-PapersOnLine*, [online] 50(1), pp.6918–6927. Available at: <<https://www.sciencedirect.com/science/article/pii/S2405896317317238>> [Accessed 8 Sep. 2018].

- Gomez, F. and Schmidhuber, J., 2005. Evolving Modular Fast-Weight Networks for Control. [online] pp.383–389. Available at: <[https://link.springer.com/chapter/10.1007/11550907\\_61](https://link.springer.com/chapter/10.1007/11550907_61)> [Accessed 4 Nov. 2018].
- Gopakumar, A.M., Balachandran, P. V., Xue, D., Gubernatis, J.E. and Lookman, T., 2018. Multi-objective Optimization for Materials Discovery via Adaptive Design. Scientific Reports, [online] 8(1), p.3738. Available at: <<http://www.nature.com/articles/s41598-018-21936-3>> [Accessed 8 Sep. 2018].
- Governo Do Estado De São Paulo, 2018. CETESB - Companhia Ambiental do Estado de São Paulo. [online] Available at: <<https://cetesb.sp.gov.br/>> [Accessed 16 Sep. 2018].
- Governo do Estado de São Paulo | Eleições 2018. [online] Available at: <<http://www.saopaulo.sp.gov.br/>> [Accessed 21 Nov. 2018].
- Grand Round Table | Daily Patient Huddle Software | Automate Chart Review. 2018. [online] Available at: <<http://www.grandroundtable.com/>> [Accessed 4 Dec. 2018].
- Gross, E., 2016. On the Bellman's principle of optimality. Physica A: Statistical Mechanics and its Applications, [online] 462, pp.217–221. Available at: <<https://www.sciencedirect.com/science/article/pii/S037843711630351X>> [Accessed 9 Sep. 2018].
- Güldal, V., Tongal, H., Güldal, V. and Tongal, H., 2010. Comparison of Recurrent Neural Network, Adaptive Neuro-Fuzzy Inference System and Stochastic Models in Engirdir Lake Level Forecasting. Water Resour Manage, [online] 24, pp.105–128. Available at: <<https://search.proquest.com/docview/216290285?pq-origsite=gscholar>> [Accessed 5 Dec. 2018].
- Haiyunnisa, T., Alam, H.S. and Salim, T.I., 2017. Design and Implementation of Fuzzy Logic Control System for Water Quality Control. pp.98–102.
- Hämäläinen, R.P. and Mäntysaari, J., 2001. A dynamic interval goal programming approach to the regulation of a lake-river system. Journal of Multi-Criteria Decision Analysis, [online] 10(2), pp.75–86. Available at: <<http://doi.wiley.com/10.1002/mcda.290>> [Accessed 9 Sep. 2018].
- Hämäläinen, R.P. and Mäntysaari, J., 2002. Dynamic multi-objective heating optimization. European Journal of Operational Research, [online] 142(1), pp.1–15. Available at: <<https://www.sciencedirect.com/science/article/pii/S037722170100282X>> [Accessed 9 Sep. 2018].
- Hansen, M.P., Hansen, M.P. and Jaskiewicz, A., 1998. Evaluating the Quality of Approximations to the Non-Dominated Set. [online] Available at: <<http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.35.5279>> [Accessed 12 Nov. 2018].
- Hasan, M.M., Abu-Hassan, K., Khin Lwin and Hossain, M.A., 2016. Reversible decision support system: Minimising cognitive dissonance in multi-criteria based complex system using fuzzy analytic hierarchy process. In: 2016 8th Computer Science and Electronic Engineering (CEECE). [online] IEEE, pp.210–215. Available at: <<http://ieeexplore.ieee.org/document/7835915/>> [Accessed 20 Sep. 2018].

- van Hasselt, H., Guez, A. and Silver, D., 2015. Deep Reinforcement Learning with Double Q-learning. 30th AAAI Conference on Artificial Intelligence, AAAI 2016, [online] pp.2094–2100. Available at: <<http://arxiv.org/abs/1509.06461>> [Accessed 28 May 2020].
- Hauser, R.A., Eftekhari, A. and Matzinger, H.F., 2018. PCA by Determinant Optimisation has no Spurious Local Optima. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining - KDD '18. [online] New York, New York, USA: ACM Press.pp.1504–1511. Available at: <<http://dl.acm.org/citation.cfm?doid=3219819.3220069>> [Accessed 7 Dec. 2018].
- Haykin, S.S. and Simon, 1994. Neural networks: a comprehensive foundation. [online] Macmillan. Available at: <<https://dl.acm.org/citation.cfm?id=541500>> [Accessed 5 Dec. 2018].
- Hazrati, S.M., Hamzeh, A. and Hashemi, S., 2013. Using reinforcement learning to find an optimal set of features. In: Computers and Mathematics with Applications. Pergamon.pp.1892–1904.
- He, F.S., Liu, Y., Schwing, A.G. and Peng, J., 2016. Learning to Play in a Day: Faster Deep Reinforcement Learning by Optimality Tightening. [online] Available at: <<http://arxiv.org/abs/1611.01606>> [Accessed 9 Sep. 2018].
- How Will Artificial Intelligence And Machine Learning Impact Cyber Security?. 2018. [online] Available at: <<https://www.forbes.com/sites/quora/2018/02/15/how-will-artificial-intelligence-and-machine-learning-impact-cyber-security/#4796fa3e6147>> [Accessed 4 Dec. 2018].
- Hein, D., Hentschel, A., Runkler, T.A. and Udluft, S., 2016. Reinforcement Learning with Particle Swarm Optimization Policy (PSO-P) in Continuous State and Action Spaces. International Journal of Swarm Intelligence Research, [online] 7. Available at: <<https://pdfs.semanticscholar.org/b51c/a3ba66d5a7fd01471a30f165831ca8da4641.pdf>> [Accessed 11 Nov. 2018].
- Helbig, M., Deb, K. and Engelbrecht, A., 2016. Key challenges and future directions of dynamic multi-objective optimisation. In: 2016 IEEE Congress on Evolutionary Computation (CEC). [online] IEEE.pp.1256–1261. Available at: <<http://ieeexplore.ieee.org/document/7743931/>> [Accessed 20 Nov. 2018].
- Helbig, M. and Engelbrecht, A.P., 2013a. Analysing the performance of dynamic multi-objective optimisation algorithms. In: 2013 IEEE Congress on Evolutionary Computation. [online] IEEE.pp.1531–1539. Available at: <<http://ieeexplore.ieee.org/document/6557744/>> [Accessed 17 Nov. 2018].
- Helbig, M. and Engelbrecht, A.P., 2013b. Benchmarks for dynamic multi-objective optimisation. In: 2013 IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments (CIDUE). [online] IEEE.pp.84–91. Available at: <<http://ieeexplore.ieee.org/document/6595776/>> [Accessed 20 Nov. 2018].
- Helbig, M. and Engelbrecht, A.P., 2013c. Performance Measures for Dynamic Multi-objective Optimisation Algorithms. [online] Available at: <[https://repository.up.ac.za/bitstream/handle/2263/33211/Helbig\\_Performance\\_2013.pdf?sequence=1](https://repository.up.ac.za/bitstream/handle/2263/33211/Helbig_Performance_2013.pdf?sequence=1)> [Accessed 12 Nov. 2018].

- Helbig, M. and Engelbrecht, A.P., 2014a. Benchmarks for dynamic multi-objective optimisation algorithms. *ACM Computing Surveys*, [online] 46(3), pp.1–39. Available at: <<http://dl.acm.org/citation.cfm?doid=2578702.2517649>> [Accessed 5 Dec. 2018].
- Helbig, M. and Engelbrecht, A.P., 2014b. Population-based metaheuristics for continuous boundary-constrained dynamic multi-objective optimisation problems. *Swarm and Evolutionary Computation*, [online] 14, pp.31–47. Available at: <<https://www.sciencedirect.com/science/article/pii/S2210650213000539>> [Accessed 9 Sep. 2018].
- Hendry, A.P., Gotanda, K.M. and Svensson, E.I., 2017. Human influences on evolution, and the ecological and societal consequences. *Philosophical transactions of the Royal Society of London. Series B, Biological sciences*, [online] 372(1712). Available at: <<http://www.ncbi.nlm.nih.gov/pubmed/27920373>> [Accessed 5 Dec. 2018].
- Herrmann, M., 2015. RL 16: Model-based RL and Multi-Objective Reinforcement Learning. [online] Available at: <<http://www.inf.ed.ac.uk/teaching/courses/rl/slides15/rl16.pdf>> [Accessed 4 Dec. 2018].
- Hoverman, J.T. and Johnson, P.T.J., 2012. Ponds and Lakes: A Journey Through the Life Aquatic | Learn Science at Scitable. *Nature Education Knowledge*, [online] p.3(6):17. Available at: <<https://www.nature.com/scitable/knowledge/library/ponds-and-lakes-a-journey-through-the-25982495>> [Accessed 9 Sep. 2018].
- Hull, V., Parrella, L. and Falcucci, M., 2008. Modelling dissolved oxygen dynamics in coastal lagoons. *Ecological Modelling*, [online] 211(3–4), pp.468–480. Available at: <<https://www.sciencedirect.com/science/article/pii/S0304380007005029>> [Accessed 5 Dec. 2018].
- Huttschenreuter, A.K., Bosman, P.A.N. and La Poutré, H., 2009. Evolutionary Multiobjective Optimization for Dynamic Hospital Resource Management. [online] Springer, Berlin, Heidelberg. pp.320–334. Available at: <[http://link.springer.com/10.1007/978-3-642-01020-0\\_27](http://link.springer.com/10.1007/978-3-642-01020-0_27)> [Accessed 9 Sep. 2018].
- IBM Watson. 2018. [online] Available at: <<https://www.ibm.com/watson/>> [Accessed 4 Dec. 2018].
- Idris, I., 2015. NumPy: beginner's guide : build efficient, high-speed programs using the high-performance NumPy mathematical library. [online] Available at: <<https://www.oreilly.com/library/view/numpy-beginners/9781785281969/>> [Accessed 21 Nov. 2018].
- Imani, M., Pascale, F., Kapogiannis, G. and Jones, K., 2016. Hospital resilience-informed decision making: uncertainties and interdependencies. [online] Available at: <<https://arro.anglia.ac.uk/700165/>> [Accessed 15 Dec. 2018].
- Isikdogan, F., Bovik, A.C. and Passalacqua, P., 2017. Surface water mapping by deep learning. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 10(11), pp.4909–4918.
- Jaderberg, M., Mnih, V., Czarnecki, W.M., Schaul, T., Leibo, J.Z., Silver, D. and Kavukcuoglu, K., 2016. Reinforcement Learning with Unsupervised Auxiliary Tasks. [online] Available at: <<http://arxiv.org/abs/1611.05397>> [Accessed 8 Sep. 2018].

- Jaimes, A.L. and Coello, C.A.C., 2015. Many-objective problems: Challenges and methods. In: Springer Handbook of Computational Intelligence. Springer Berlin Heidelberg. pp.1033–1046.
- Janssen, M., van der Voort, H. and Wahyudi, A., 2017. Factors influencing big data decision-making quality. *Journal of Business Research*, [online] 70, pp.338–345. Available at: <<https://linkinghub.elsevier.com/retrieve/pii/S0148296316304945>> [Accessed 4 Dec. 2018].
- Jarrahi, M.H., 2018. Artificial intelligence and the future of work: Human-AI symbiosis in organizational decision making. *Business Horizons*, 61(4), pp.577–586.
- Ji, X., Shang, X., Dahlgren, R.A. and Zhang, M., 2017. Prediction of dissolved oxygen concentration in hypoxic river systems using support vector machine: a case study of Wen-Rui Tang River, China. *Environmental Science and Pollution Research*, [online] 24(19), pp.16062–16076. Available at: <<http://www.ncbi.nlm.nih.gov/pubmed/28537025>> [Accessed 9 Sep. 2018].
- Jiang, S. and Yang, S., 2014. A benchmark generator for dynamic multi-objective optimization problems. In: 2014 14th UK Workshop on Computational Intelligence (UKCI). [online] IEEE. pp.1–8. Available at: <<http://ieeexplore.ieee.org/document/6930171/>> [Accessed 5 Dec. 2018].
- Jiang, S., Yang, S., Yao, X., Tan, K.C. and Kaiser, M., 2018. Benchmark Problems for CEC2018 Competition on Dynamic Multiobjective Optimisation. pp.1–18.
- Jordan, M.I. and Mitchell, T.M., 2015. Machine learning: Trends, perspectives, and prospects. *Science*, [online] 349(6245), pp.255–260. Available at: <<http://science.sciencemag.org/content/349/6245/255>> [Accessed 8 Dec. 2018].
- Julian Togelius, 2007. Optimization, Imitation and Innovation: Computational Intelligence and Games. [online] Available at: <<http://julian.togelius.com/thesis.pdf>> [Accessed 17 Feb. 2020].
- Justesen, N., Torrado, R.R., Bontrager, P., Khalifa, A., Togelius, J. and Risi, S., 2018. Illuminating Generalization in Deep Reinforcement Learning through Procedural Level Generation. [online] Available at: <<http://arxiv.org/abs/1806.10729>> [Accessed 4 Dec. 2018].
- Justin Francis, 2017. Introduction to reinforcement learning and OpenAI Gym - O'Reilly Media. [online] Available at: <<https://www.oreilly.com/learning/introduction-to-reinforcement-learning-and-openai-gym>> [Accessed 7 Dec. 2018].
- Kalin, L., Isik, S., Schoonover, J.E. and Lockaby, B.G., 2010. Predicting water quality in unmonitored watersheds using artificial neural networks. *Journal of environmental quality*, [online] 39(4), pp.1429–40. Available at: <<http://www.ncbi.nlm.nih.gov/pubmed/20830930>> [Accessed 5 Dec. 2018].
- Kantas, N., 2009. Sequential Decision Making in General State Space models. (February), pp.1–239.

- Kapoor, S., 2018. Multi-Agent Reinforcement Learning: A Report on Challenges and Approaches. [online] Available at: <<https://arxiv.org/pdf/1807.09427.pdf>> [Accessed 11 Nov. 2018].
- Kasey Panetta, 2017. Gartner Top 10 Strategic Technology Trends for 2018 - Smarter With Gartner. [online] Gartner. Available at: <<https://www.gartner.com/smarterwithgartner/gartner-top-10-strategic-technology-trends-for-2018/>> [Accessed 4 Dec. 2018].
- Khalil, B., Adamowski, J., Abdin, A.E. and Eizeldin, M., 2014. Estimation of Water Quality Characteristics at Ungauged Sites using Multiple Linear Regression and Canonical Correlation Analysis. In: 2014 ASABE International Meeting. [online] American Society of Agricultural and Biological Engineers.pp.1–9. Available at: <<http://elibrary.asabe.org/abstract.asp?aid=44492&t=3&dabs=Y&redir=&redirType=>>> [Accessed 5 Dec. 2018].
- Kim, K. and Park, K.S., 1990. Ranking fuzzy numbers with index of optimism. Fuzzy Sets and Systems, [online] 35(2), pp.143–150. Available at: <<https://www.sciencedirect.com/science/article/pii/016501149090189D>> [Accessed 9 Oct. 2018].
- King, J.L. and Grudin, J., 2016. Will Computers Put Us Out of Work? Computer, 49(5), pp.82–85.
- Kingma, D.P. and Ba, J., 2014. Adam: A Method for Stochastic Optimization. [online] Available at: <<http://arxiv.org/abs/1412.6980>> [Accessed 23 Sep. 2018].
- Klir, G.J. and Yuan, B., 1995. Fuzzy sets and fuzzy logic: theory and applications. Prentice Hall PTR.
- Kober, J., Bagnell, J.A. and Peters, J., 2013. Reinforcement learning in robotics: A survey. The International Journal of Robotics Research, [online] 32(11), pp.1238–1274. Available at: <<http://journals.sagepub.com/doi/10.1177/0278364913495721>> [Accessed 8 Sep. 2018].
- Koenig, S. and Simmons, R.G., 1993. Complexity Analysis of Real-Time Reinforcement Learning. [online] Available at: <<https://www.aaai.org/Papers/AAAI/1993/AAAI93-016.pdf>> [Accessed 10 May 2020].
- Koo, W.T., Goh, C.K. and Tan, K.C., 2010. A predictive gradient strategy for multiobjective evolutionary algorithms in a fast changing environment. Memetic Computing, [online] 2(2), pp.87–110. Available at: <<http://link.springer.com/10.1007/s12293-009-0026-7>> [Accessed 10 Oct. 2018].
- Koutník, J., Cuccu, G., Schmidhuber, J. and Gomez, F., 2013. Evolving large-scale neural networks for vision-based reinforcement learning. In: Proceeding of the fifteenth annual conference on Genetic and evolutionary computation conference - GECCO '13. [online] New York, New York, USA: ACM Press.p.1061. Available at: <<http://dl.acm.org/citation.cfm?doid=2463372.2463509>> [Accessed 8 Sep. 2018].
- Krizhevsky, A., Sutskever, I. and Hinton, G.E., 2012. ImageNet classification with deep convolutional neural networks. Proceedings of the 25th International Conference on



- Neural Information Processing Systems - Volume 1, Available at: <https://dl.acm.org/citation.cfm?id=2999257> [Accessed 12 Nov. 2018].
- Lam T. Bui, Branke, J. and Abbass, H.A., 2005. Multiobjective optimization for dynamic environments. In: 2005 IEEE Congress on Evolutionary Computation. [online] IEEE, pp.2349–2356. Available at: <http://ieeexplore.ieee.org/document/1554987/> [Accessed 19 Nov. 2018].
- Lampinen, J., Lampinen, J. and Zelinka, I., 1999. Mixed Integer-Discrete-Continuous Optimization By Differential Evolution - Part 2: a practical example. CZECH REPUBLIC. BRNO UNIVERSITY OF TECHNOLOGY, [online] pp.77--81. Available at: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.35.7014> [Accessed 15 Dec. 2018].
- Laura Schneider, 2018. A History of the Technology Industry. [online] Available at: <https://www.thebalancecareers.com/a-history-of-the-technology-industry-2071506> [Accessed 8 Dec. 2018].
- Laurens van der Maaten, 2018. t-SNE – Laurens van der Maaten. [online] Available at: <https://lvdmaaten.github.io/tsne/> [Accessed 17 Nov. 2018].
- Le, Q. V, Ngiam, J., Coates, A., Lahiri, A., Prochnow, B. and Ng, A.Y., 2011. On Optimization Methods for Deep Learning. [online] Available at: <https://ai.stanford.edu/~ang/papers/icml11-OptimizationForDeepLearning.pdf> [Accessed 8 Dec. 2018].
- LeCun, Y., Bengio, Y. and Hinton, G., 2015. Deep learning. Nature, [online] 521(7553), pp.436–444. Available at: <http://www.nature.com/articles/nature14539> [Accessed 1 Sep. 2018].
- Leger, S., Zwanenburg, A., Pilz, K., Lohaus, F., Linge, A., Zöphel, K., Kotzerke, J., Schreiber, A., Tinhofer, I., Budach, V., Sak, A., Stuschke, M., Balermipas, P., Rödel, C., Ganswindt, U., Belka, C., Pigorsch, S., Combs, S.E., Mönnich, D., Zips, D., Krause, M., Baumann, M., Troost, E.G.C., Löck, S. and Richter, C., 2017. A comparative study of machine learning methods for time-to-event survival data for radiomics risk modelling. Scientific Reports, [online] 7(1), p.13206. Available at: <http://www.nature.com/articles/s41598-017-13448-3> [Accessed 1 Sep. 2018].
- Levine, S. and Koltun, V., 2013. Guided Policy Search. [online] Available at: [https://graphics.stanford.edu/projects/gpspaper/gps\\_full.pdf](https://graphics.stanford.edu/projects/gpspaper/gps_full.pdf) [Accessed 10 Nov. 2018].
- Levy, A., Platt, R. and Saenko, K., 2018. Hierarchical Reinforcement Learning with Hindsight. [online] Available at: <http://arxiv.org/abs/1805.08180> [Accessed 12 Nov. 2018].
- Li, C., 2016. An Efficient Benchmark Generator for Dynamic Optimization Problems. [online] Springer, Singapore, pp.60–72. Available at: [http://link.springer.com/10.1007/978-981-10-3614-9\\_8](http://link.springer.com/10.1007/978-981-10-3614-9_8) [Accessed 5 Dec. 2018].
- Li, C., Yang, S., Nguyen, T.T., Yu, E.L., Yao, X., Jin, Y., Beyer, H.-G. and Suganthan, P.N., 2008. Benchmark Generator for CEC'2009 Competition on Dynamic Optimization. [online] Available at:

<<http://www.cs.le.ac.uk/people/syang/ECiDUE/DBG.tar.gz> and <http://www.ntu.edu.sg/home/epsnugan/DBG.tar.gz> respectively. The <http://www.cs.bham.ac.uk/research/projects/ecb/>> [Accessed 5 Dec. 2018].

Li, E., Zeng, L., Zhou, Z. and Chen, X., 2019. Edge AI: On-Demand Accelerating Deep Neural Network Inference via Edge Computing. [online] Available at: <<http://arxiv.org/abs/1910.05316>> [Accessed 5 Mar. 2020].

Li, H., Zhang, Q. and Deng, J., 2017. Biased Multiobjective Optimization and Decomposition Algorithm. *IEEE Transactions on Cybernetics*, [online] 47(1), pp.52–66. Available at: <<http://ieeexplore.ieee.org/document/7397980/>> [Accessed 8 Sep. 2018].

Li, J., Liu, Z., He, C., Yue, H. and Gou, S., 2017. Water shortages raised a legitimate concern over the sustainable development of the drylands of northern China: Evidence from the water stress index. *Science of The Total Environment*, [online] 590–591, pp.739–750. Available at: <<http://www.ncbi.nlm.nih.gov/pubmed/28284646>> [Accessed 8 Sep. 2018].

Li, Y., 2017. Deep Reinforcement Learning: An Overview. [online] pp.1–70. Available at: <<http://arxiv.org/abs/1701.07274>>.

Li, Z., Chen, H., Xie, Z., Chen, C. and Sallam, A., 2014. Dynamic multiobjective optimization algorithm based on average distance linear prediction model. *TheScientificWorldJournal*, [online] 2014, p.389742. Available at: <<http://www.ncbi.nlm.nih.gov/pubmed/24616625>> [Accessed 9 Sep. 2018].

Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D. and Wierstra, D., 2015. Continuous control with deep reinforcement learning. [online] Available at: <<http://arxiv.org/abs/1509.02971>> [Accessed 12 Nov. 2018].

Lima, G.N. de, Lombardo, M.A. and Magaña Rueda, V.O., 2018. Data on the volumes of water stored in the reservoirs supplying the Metropolitan Area of Sao Paulo (2003–2015). *Data in Brief*, [online] 19, pp.409–412. Available at: <<https://linkinghub.elsevier.com/retrieve/pii/S2352340918305559>> [Accessed 8 Sep. 2018].

Lima, G.N. de, Lombardo, M.A. and Magaña, V., 2018. Urban water supply and the changes in the precipitation patterns in the metropolitan area of São Paulo – Brazil. *Applied Geography*, [online] 94, pp.223–229. Available at: <<https://www.sciencedirect.com/science/article/pii/S0143622817304769>> [Accessed 8 Sep. 2018].

Lin, Z., Zhao, T., Yang, G. and Zhang, L., 2017. Episodic Memory Deep Q-Networks. [online] Available at: <<https://www.ijcai.org/proceedings/2018/0337.pdf>> [Accessed 4 Dec. 2018].

Lindberg, R.H., Östman, M., Olofsson, U., Grabic, R. and Fick, J., 2014. Occurrence and behaviour of 105 active pharmaceutical ingredients in sewage waters of a municipal sewer collection system. *Water Research*, [online] 58, pp.221–229. Available at: <<http://www.ncbi.nlm.nih.gov/pubmed/24768701>> [Accessed 8 Sep. 2018].

Liu, C.-A., Wang, Y. and Ren, A., 2015. New Dynamic Multi-Objective Constrained Optimization Evolutionary Algorithm. *Asia-Pacific Journal of Operational Research*,

- [online] 32(05), p.1550036. Available at: <<http://www.worldscientific.com/doi/abs/10.1142/S0217595915500360>> [Accessed 10 Oct. 2018].
- Liu, C., Xu, X. and Hu, D., 2015. Multiobjective reinforcement learning: A comprehensive overview. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(3), pp.385–398.
- Liu, S., Zhang, G. and Wang, L., 2018. IoT-enabled Dynamic Optimisation for Sustainable Reverse Logistics. *Procedia CIRP*, [online] 69, pp.662–667. Available at: <<https://www.sciencedirect.com/science/article/pii/S2212827117308648>> [Accessed 11 Nov. 2018].
- Liu, Y., Logan, B., Liu, N., Xu, Z., Tang, J. and Wang, Y., 2017. Deep Reinforcement Learning for Dynamic Treatment Regimes on Medical Registry Data. *Healthcare informatics: the business magazine for information and communication systems*, [online] 2017, pp.380–385. Available at: <<http://www.ncbi.nlm.nih.gov/pubmed/29556119>> [Accessed 12 Nov. 2018].
- Lizotte, D., Bowling, M. and Murphy, S., 2010. Efficient reinforcement learning with multiple reward functions for randomized controlled trial analysis. *Proceedings of the 27th International Conference on Machine Learning*. [online] Available at: <<https://methodology.psu.edu/node/1421>> [Accessed 9 Sep. 2018].
- Lizotte, D.J. and Elbert, E.B., 2016. Multi-Objective Markov Decision Processes for Data-Driven Decision Support. *Journal of Machine Learning Research*, [online] 17(211), pp.1–28. Available at: <<http://www.jmlr.org/papers/v17/15-252.html>> [Accessed 19 Nov. 2018].
- Lopes Silva, M.A., de Souza, S.R., Freitas Souza, M.J. and Bazzan, A.L.C., 2019. A reinforcement learning-based multi-agent framework applied for solving routing and scheduling problems. *Expert Systems with Applications*, 131, pp.148–171.
- Lu, Y., Song, S., Wang, R., Liu, Z., Meng, J., Sweetman, A.J., Jenkins, A., Ferrier, R.C., Li, H., Luo, W. and Wang, T., 2015. Impacts of soil and water pollution on food safety and health risks in China. *Environment International*, [online] 77, pp.5–15. Available at: <<http://www.ncbi.nlm.nih.gov/pubmed/25603422>> [Accessed 5 Dec. 2018].
- Luke Dormehl, 2018. MIT Trained an A.I. With Reddit, and it Became a Psychopath | Digital Trends. [online] Available at: <<https://www.digitaltrends.com/cool-tech/ai-norman-mit-project/>> [Accessed 4 Dec. 2018].
- Lwin, K., Qu, R. and Kendall, G., 2014. A learning-guided multi-objective evolutionary algorithm for constrained portfolio optimization. *Applied Soft Computing*, Available at: <<http://linkinghub.elsevier.com/retrieve/pii/S1568494614003913>>.
- Maalawi, K., 2011. Special Issues on Design Optimization of Wind Turbine Structures. [online] Available at: <[www.intechopen.com](http://www.intechopen.com)> [Accessed 9 Sep. 2018].
- Machado, M.C., Bellemare, M.G., Talvitie, E., Veness, J., Hausknecht, M. and Bowling, M., 2017. Revisiting the Arcade Learning Environment: Evaluation Protocols and Open Problems for General Agents. [online] Available at: <<http://arxiv.org/abs/1709.06009>> [Accessed 7 Dec. 2018].

- Machine Learning on AWS. 2018. [online] Available at:  
<<https://aws.amazon.com/machine-learning/>> [Accessed 4 Dec. 2018].
- Mark Lake, 2001. Example 2: Hill-Climbing and the Local Optimum Problem. [online] Available at:  
<<http://www.ucl.ac.uk/~tcnmar/simulation/magical/examples/node2.html>> [Accessed 7 Dec. 2018].
- Martin, M., 2011. Searching for optimal policies I: Bellman equations and optimal policies, Lecture Notes, Learning in Agents and Multiagents Systems, Universitat politècnica de Catalunya, delivered in Autumn 2011. Available at:  
<<https://www.cs.upc.edu/~mmartin/Ag4-4x.pdf>> [Accessed 28 May 2020].
- Matt O'Brien, 2020. Pentagon adopts new ethical principles for using AI in war. [online] apnews.com. Available at:  
<[https://apnews.com/73df704904522f5a66a92bc5c4df8846?fbclid=IwAR1m8V0yeXssAANiIp3pgz4N9Hb3qWN\\_mPtPpK\\_AvWlaDLLhwXNNLVmdaHg](https://apnews.com/73df704904522f5a66a92bc5c4df8846?fbclid=IwAR1m8V0yeXssAANiIp3pgz4N9Hb3qWN_mPtPpK_AvWlaDLLhwXNNLVmdaHg)> [Accessed 5 Mar. 2020].
- Matthews, R., Hilles, M. and Pelletier, G., 2002. Determining trophic state in Lake Whatcom, Washington (USA), a soft water lake exhibiting seasonal nitrogen limitation. *Hydrobiologia*, [online] 468(1/3), pp.107–121. Available at:  
<<http://link.springer.com/10.1023/A:1015288519122>> [Accessed 5 Dec. 2018].
- Mehnen, J., Wagner, T. and Rudolph, G., 2006. Evolutionary Optimization of Dynamic Multiobjective Functions. Available at:  
<<https://www.semanticscholar.org/paper/Evolutionary-Optimization-of-Dynamic-Multiobjective-Mehnen-Wagner/35e80a2234c2782b34812654c0dd39a84c3f5eae>> [Accessed 11 Sep. 2018].
- Meisel, S., Grimme, C., Bossek, J., Wölck, M., Rudolph, G. and Trautmann, H., 2015. Evaluation of a Multi-Objective EA on Benchmark Instances for Dynamic Routing of a Vehicle. [online] Available at: <<http://dx.doi.org/10.1145/2739480.2754705>> [Accessed 9 Sep. 2018].
- Mirjalili, S. and Lewis, A., 2015. Novel performance metrics for robust multi-objective optimization algorithms. *Swarm and Evolutionary Computation*, [online] 21, pp.1–23. Available at: <<https://www.sciencedirect.com/science/article/pii/S2210650214000777>> [Accessed 17 Nov. 2018].
- Mirowski, P., Pascanu, R., Viola, F., Soyer, H., Ballard, A.J., Banino, A., Denil, M., Goroshin, R., Sifre, L., Kavukcuoglu, K., Kumaran, D. and Hadsell, R., 2016. Learning to Navigate in Complex Environments. [online] Available at:  
<<http://arxiv.org/abs/1611.03673>> [Accessed 8 Sep. 2018].
- MIT Technology Review, 2018. 10 Breakthrough Technologies 2017 - MIT Technology Review. [online] Available at:  
<<https://www.technologyreview.com/lists/technologies/2017/>> [Accessed 8 Sep. 2018].
- Mitchell, T.M., 2006. Markov Decision Processes and Reinforcement Learning. [online] Available at: <[https://www.cs.cmu.edu/~epxing/Class/10701-06f/lectures/MDPs\\_RL\\_11\\_30\\_06.pdf](https://www.cs.cmu.edu/~epxing/Class/10701-06f/lectures/MDPs_RL_11_30_06.pdf)> [Accessed 21 Nov. 2018].

- Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D. and Riedmiller, M., 2013. Playing Atari with Deep Reinforcement Learning. [online] Available at: <<http://arxiv.org/abs/1312.5602>> [Accessed 28 May 2020].
- Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S. and Hassabis, D., 2015. Human-level control through deep reinforcement learning. *Nature*, [online] 518. Available at: <<https://www.nature.com/articles/nature14236.pdf>> [Accessed 19 Jul. 2018].
- MobileODT | The Smart Mobile Colposcope. 2018. [online] Available at: <<https://www.mobileodt.com/>> [Accessed 4 Dec. 2018].
- Moffaert, K. Van, Drugan, M.M. and Nowé, A., 2014. Learning Sets of Pareto Optimal Policies. [online] Available at: <<https://www.researchgate.net/publication/260487660>> [Accessed 11 Nov. 2018].
- Moffaert, K. Van and Nowé, A., 2014. Multi-Objective Reinforcement Learning using Sets of Pareto Dominating Policies. *Journal of Machine Learning Research*, [online] 15, pp.3663–3692. Available at: <<http://jmlr.org/papers/volume15/vanmoffaert14a/vanmoffaert14a.pdf>> [Accessed 19 Jul. 2018].
- Morales-Enciso, S. and Branke, J., 2014. Tracking global optima in dynamic environments with efficient global optimization. *European Journal of Operational Research*, [online] 242, pp.744–755. Available at: <<http://dx.doi.org/10.1016/j.ejor.2014.11.028>> [Accessed 17 Nov. 2018].
- Moritz, R.L. V, Reich, E., Schwarz, M., Bernt, M. and Middendorf, M., 2014. Refined ranking relations for selection of solutions in multi objective metaheuristics. *European Journal of Operational Research*, [online] 243, pp.454–464. Available at: <<http://dx.doi.org/10.1016/j.ejor.2014.10.044>> [Accessed 17 Nov. 2018].
- Mossalam, H., Assael, Y.M., Roijers, D.M. and Whiteson, S., 2016. Multi-Objective Deep Reinforcement Learning. [online] Available at: <<https://arxiv.org/pdf/1610.02707.pdf>> [Accessed 20 Jun. 2017].
- Mugume, S.N., Diao, K., Astaraie-Imani, M., Fu, G., Farmani, R. and Butler, D., 2015. Enhancing resilience in urban water systems for future cities. *Water Science & Technology: Water Supply*. [online] Available at: <<https://core.ac.uk/download/pdf/43095382.pdf>> [Accessed 13 Dec. 2018].
- Multi-criteria analysis: a manual. 2009. [online] Available at: <[www.communities.gov.ukcommunity,opportunity,prosperity](http://www.communities.gov.uk/community,opportunity,prosperity)> [Accessed 14 Nov. 2018].
- Muruganantham, A., Tan, K.C. and Vadakkepat, P., 2016. Solving the IEEE CEC 2015 Dynamic Benchmark Problems Using Kalman Filter Based Dynamic Multiobjective Evolutionary Algorithm. [online] *Springer, Cham*.pp.239–252. Available at: <[http://link.springer.com/10.1007/978-3-319-27000-5\\_20](http://link.springer.com/10.1007/978-3-319-27000-5_20)> [Accessed 9 Sep. 2018].

- Nachum, O., Brain, G., Gu, S., Lee, H. and Levine, S., 2018. Data-Efficient Hierarchical Reinforcement Learning. [online] Available at: <<https://sites.google.com/view/efficient-hrl>> [Accessed 11 Nov. 2018].
- Najah, A., El-Shafie, A., Karim, O.A. and El-Shafie, A.H., 2013. Application of artificial neural networks for water quality prediction. *Neural Computing and Applications*, [online] 22(S1), pp.187–201. Available at: <<http://link.springer.com/10.1007/s00521-012-0940-3>> [Accessed 5 Dec. 2018].
- Nandy, A. and Biswas, M., 2018. Reinforcement learning: with Open AI, TensorFlow and Keras using Python.
- Narasimhan, K., Yala, A. and Barzilay, R., 2016. Improving Information Extraction by Acquiring External Evidence with Reinforcement Learning. [online] Available at: <<http://people.csail.mit.edu>> [Accessed 8 Sep. 2018].
- Natarajan, S. and Tadepalli, P., 2005. Dynamic preferences in multi-criteria reinforcement learning. In: *Proceedings of the 22nd international conference on Machine learning - ICML '05*. [online] New York, New York, USA: ACM Press. pp.601–608. Available at: <<http://portal.acm.org/citation.cfm?doid=1102351.1102427>> [Accessed 8 Sep. 2018].
- Nedjah, N. and Mourelle, L. de M., 2015. Evolutionary multi-objective optimisation: a survey. *International Journal of Bio-Inspired Computation*, [online] 7(1), p.1. Available at: <<http://www.inderscience.com/link.php?id=67991>> [Accessed 7 Dec. 2018].
- Ng, A.Y., 2003. Shaping and policy search in reinforcement learning. *ProQuest Dissertations and Theses*, [online] 3105322, pp.155-155 p. Available at: <[http://ezproxy.net.ucf.edu/login?url=http://search.proquest.com/docview/305341681?accountid=10003%5Chttp://sfx.fcla.edu/ucf?url\\_ver=Z39.88-2004&rft\\_val\\_fmt=info:ofi/fmt:kev:mtx:dissertation&genre=dissertations+&+theses&sid=ProQ:ProQuest+Dissertations+&+T](http://ezproxy.net.ucf.edu/login?url=http://search.proquest.com/docview/305341681?accountid=10003%5Chttp://sfx.fcla.edu/ucf?url_ver=Z39.88-2004&rft_val_fmt=info:ofi/fmt:kev:mtx:dissertation&genre=dissertations+&+theses&sid=ProQ:ProQuest+Dissertations+&+T)>.
- Nguyen, T.T., 2011. Continuous dynamic optimisation using evolutionary algorithms. [online] Available at: <<http://etheses.bham.ac.uk/1296/>> [Accessed 4 Nov. 2018].
- Nguyen, T.T., 2018. A Multi-Objective Deep Reinforcement Learning Framework. [online] Available at: <<http://arxiv.org/abs/1803.02965>> [Accessed 7 Dec. 2018].
- Nguyen, T.T., Yang, S., Branke, J. and Yao, X., 2013. Evolutionary Dynamic Optimization: Methodologies. [online] Springer, Berlin, Heidelberg. pp.39–64. Available at: <[http://link.springer.com/10.1007/978-3-642-38416-5\\_2](http://link.springer.com/10.1007/978-3-642-38416-5_2)> [Accessed 8 Dec. 2018].
- Nguyen, T.T., Yang, Z. and Bonsall, S., 2012. Dynamic time-linkage problems - The challenges. In: *2012 IEEE RIVF International Conference on Computing and Communication Technologies, Research, Innovation, and Vision for the Future, RIVF 2012*.
- Nguyen, T.T. and Yao, X., 2009. Dynamic Time-Linkage Problems Revisited. [online] Springer, Berlin, Heidelberg. pp.735–744. Available at: <[http://link.springer.com/10.1007/978-3-642-01129-0\\_83](http://link.springer.com/10.1007/978-3-642-01129-0_83)> [Accessed 15 Dec. 2018].

- Nick Routley, 2017. Visualizing the Trillion-Fold Increase in Computing Power. [online] Available at: <<https://www.visualcapitalist.com/visualizing-trillion-fold-increase-computing-power/>> [Accessed 15 Dec. 2018].
- Nogueira, T.A.R., Abreu-Junior, C.H., Alleoni, L.R.F., He, Z., Soares, M.R., Santos Vieira, C. dos, Lessa, L.G.F. and Capra, G.F., 2018. Background concentrations and quality reference values for some potentially toxic elements in soils of São Paulo State, Brazil. *Journal of Environmental Management*, [online] 221, pp.10–19. Available at: <<https://www.sciencedirect.com/science/article/pii/S0301479718305735>> [Accessed 8 Sep. 2018].
- Okabe, T., Yaochu Jin and Sendhoff, B., 2003. A critical survey of performance indices for multi-objective optimisation. In: *The 2003 Congress on Evolutionary Computation*, 2003. CEC '03. [online] IEEE.pp.878–885. Available at: <<http://ieeexplore.ieee.org/document/1299759/>> [Accessed 17 Nov. 2018].
- Oktaria, D., Suhardi and Kurniawan, N.B., 2017. Smart city services: A systematic literature review. In: *2017 International Conference on Information Technology Systems and Innovation (ICITSI)*. [online] IEEE.pp.206–213. Available at: <<http://ieeexplore.ieee.org/document/8267944/>> [Accessed 4 Dec. 2018].
- Palani, S., Liong, S.-Y. and Tkalic, P., 2008. An ANN application for water quality forecasting. *Marine Pollution Bulletin*, [online] 56(9), pp.1586–1597. Available at: <<http://www.ncbi.nlm.nih.gov/pubmed/18635240>> [Accessed 5 Dec. 2018].
- Panov, A.I., Yakovlev, K.S. and Suvorov, R., 2018. Grid Path Planning with Deep Reinforcement Learning: Preliminary Results. *Procedia Computer Science*, [online] 123, pp.347–353. Available at: <<https://www.sciencedirect.com/science/article/pii/S1877050918300553>> [Accessed 4 Dec. 2018].
- Paramita Ghosh, 2018. Deep Learning Updates: Machine Learning, Deep Reinforcement Learning, and Limitations - DATAVERSITY. [online] Available at: <<http://www.dataversity.net/deep-learning-updates-machine-learning-deep-reinforcement-learning-limitations/>> [Accessed 10 Nov. 2018].
- Parisi, S., Pirotta, M., Smacchia, N., Bascetta, L. and Restelli, M., 2014. Policy Gradient Approaches for Multi-Objective Sequential Decision Making. [online] Available at: <<https://pdfs.semanticscholar.org/presentation/c0f8/4df5d5f6cbe8b691a260f20d80aa82ab46b6.pdf>> [Accessed 4 Dec. 2018].
- Paul, 2015. Sustainable Development in Developing Countries: Ramifications of Urbanisation and Poverty | OpenPop.org. [online] Available at: <<http://www.openpop.org/?p=1054>> [Accessed 5 Dec. 2018].
- Pawara, S., Nalam, S., Mirajkar, S., Gujar, S. and Nagmoti, V., 2017. Remote monitoring of waters quality from reservoirs. In: *2017 2nd International Conference for Convergence in Technology, I2CT 2017*. Institute of Electrical and Electronics Engineers Inc.pp.503–506.
- Peek, N.B., 1999. Explicit temporal models for decision–theoretic planning of clinical management. *Artificial Intelligence in Medicine*, [online] 15(2), pp.135–154. Available

at: <<https://www.sciencedirect.com/science/article/pii/S0933365798000499>> [Accessed 8 Sep. 2018].

Perez, D., Mostaghim, S., Samothrakis, S. and Lucas, S.M., 2015. Multiobjective Monte Carlo Tree Search for Real-Time Games. *IEEE Transactions on Computational Intelligence and AI in Games*, [online] 7(4), pp.347–360. Available at: <<http://ieeexplore.ieee.org/document/6872573/>> [Accessed 8 Sep. 2018].

Perez, D., Samothrakis, S. and Lucas, S., 2013. Online and offline learning in multi-objective Monte Carlo Tree Search. In: 2013 IEEE Conference on Computational Intelligence in Games (CIG). [online] IEEE.pp.1–8. Available at: <<http://ieeexplore.ieee.org/document/6633621/>> [Accessed 17 Nov. 2018].

Piero Scaruffi, 1996. A Timeline of Artificial Intelligence. [online] Available at: <<https://www.scaruffi.com/mind/ai.html>> [Accessed 8 Dec. 2018].

Plaisted, D.A., 1984. New NP-hard and NP-complete polynomial and integer divisibility problems. *Theoretical Computer Science*, [online] 31(1–2), pp.125–138. Available at: <<https://www.sciencedirect.com/science/article/pii/0304397584901300>> [Accessed 15 Dec. 2018].

Preissner, S., Dunkel, M., Hoffmann, M.F., Preissner, S.C., Genov, N., Rong, W.W., Preissner, R. and Seeger, K., 2012. Drug Cocktail Optimization in Chemotherapy of Cancer. *PLoS ONE*, [online] 7(12), p.e51020. Available at: <<http://dx.plos.org/10.1371/journal.pone.0051020>> [Accessed 8 Sep. 2018].

Publicações e Relatórios | Águas Interiores. 2017. [online] Available at: <<https://cetesb.sp.gov.br/aguas-interiores/publicacoes-e-relatorios/>> [Accessed 20 Sep. 2018].

Python Data Analysis Library — pandas: Python Data Analysis Library. 2018. [online] Available at: <<https://pandas.pydata.org/>> [Accessed 14 Nov. 2018].

Rao, K.R., Mudaliar, R.K., Hasan, M.K. and Alam, M.K., 2017. A Perfect Mathematical Formulation of Fuzzy Transportation Problem Provides an Optimal Solution Speedily. In: *Proceedings - Asia-Pacific World Congress on Computer Science and Engineering 2016 and Asia-Pacific World Congress on Engineering 2016, APWC on CSE/APWCE 2016*.

Raquel, C. and Yao, X., 2013. Dynamic Multi-objective Optimization: A Survey of the State-of-the-Art. [online] Springer, Berlin, Heidelberg.pp.85–106. Available at: <[http://link.springer.com/10.1007/978-3-642-38416-5\\_4](http://link.springer.com/10.1007/978-3-642-38416-5_4)> [Accessed 16 Sep. 2018].

Rathod, P.M., Marathe, N. and Vidhate, A. V., 2014. A survey on Finite Automata based pattern matching techniques for network Intrusion Detection System (NIDS). In: 2014 International Conference on Advances in Electronics Computers and Communications. [online] IEEE.pp.1–5. Available at: <<http://ieeexplore.ieee.org/document/7002456/>> [Accessed 18 Dec. 2018].

Ravichandiran, S., 2018. Hands-On Reinforcement Learning with Python.



- Rojjers, D.M., Vamplew, P., Whiteson, S. and Dazeley, R., 2013. A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research*, 48, pp.67–113.
- Rondeau, L., Ruelas, R., Levrat, L. and Lamotte, M., 1997. A defuzzification method respecting the fuzzification. *Fuzzy Sets and Systems*, [online] 86(3), pp.311–320. Available at: <<https://www.sciencedirect.com/science/article/pii/S0165011495003991>> [Accessed 9 Oct. 2018].
- Ruben, 2016. Reinforcement Learning and DQN, learning to play from pixels - Ruben Fiszal's website. [online] Available at: <<https://rubenfiszal.github.io/posts/rl4j/2016-08-24-Reinforcement-Learning-and-DQN.html>> [Accessed 4 Dec. 2018].
- Ruder, S., 2016. An overview of gradient descent optimization algorithms. [online] Available at: <<http://arxiv.org/abs/1609.04747>> [Accessed 6 Dec. 2018].
- Ruiz-Montiel, M., Mandow, L. and Pérez-de-la-Cruz, J.L., 2017. A temporal difference method for multi-objective reinforcement learning. *Neurocomputing*, [online] 263, pp.15–25. Available at: <<http://dx.doi.org/10.1016/j.neucom.2016.10.100>>.
- Sadeghi, S.H. and Hazbavi, Z., 2017. Spatiotemporal variation of watershed health propensity through reliability-resilience-vulnerability based drought index (case study: Shazand Watershed in Iran). *Science of The Total Environment*, [online] 587–588, pp.168–176. Available at: <<http://www.ncbi.nlm.nih.gov/pubmed/28249754>> [Accessed 5 Dec. 2018].
- Salimans, T., Ho, J., Chen, X., Sidor, S. and Sutskever, I., 2017. Evolution Strategies as a Scalable Alternative to Reinforcement Learning. [online] Available at: <<http://arxiv.org/abs/1703.03864>> [Accessed 4 Nov. 2018].
- Samonji, H. and Watanabe, S., 2017. Escaping from Local Optima and Convergence Mechanisms Based on Search History in Evolutionary Multi-criterion Optimization. *Transactions of the Japanese Society for Artificial Intelligence*, [online] 32(3), p.E-GB1\_1-12. Available at: <[https://www.jstage.jst.go.jp/article/tjsai/32/3/32\\_E-GB1/\\_article/-char/ja/](https://www.jstage.jst.go.jp/article/tjsai/32/3/32_E-GB1/_article/-char/ja/)> [Accessed 17 Nov. 2018].
- Sarkar, A. and Pandey, P., 2015. River Water Quality Modelling Using Artificial Neural Network Technique. *Aquatic Procedia*, [online] 4, pp.1070–1077. Available at: <<https://www.sciencedirect.com/science/article/pii/S2214241X15001364>> [Accessed 5 Dec. 2018].
- Sarkar, D., Contal, E., Vayatis, N. and Dias, F., 2015. A Machine Learning Approach to the Analysis of Wave Energy Converters. In: Volume 9: Ocean Renewable Energy. [online] ASME.p.V009T09A019. Available at: <<http://proceedings.asmedigitalcollection.asme.org/proceeding.aspx?doi=10.1115/OMA E2015-41523>> [Accessed 9 Sep. 2018].
- SauceCat, 2017. Introducing pydq – Towards Data Science. [online] Available at: <<https://towardsdatascience.com/introducing-pydq-7f23d04076b3>> [Accessed 17 Nov. 2018].
- Scarcello, F., 2019. Artificial Intelligence. *Encyclopedia of Bioinformatics and Computational Biology*, [online] pp.287–293. Available at:

<<https://www.sciencedirect.com/science/article/pii/B9780128096338203269>>  
[Accessed 16 Nov. 2018].

Schaul, T., Quan, J., Antonoglou, I. and Silver, D., 2015. Prioritized Experience Replay. [online] Available at: <<http://arxiv.org/abs/1511.05952>> [Accessed 4 Dec. 2018].

Schmidhuber, J., 2015. Deep learning in neural networks: An overview. Neural Networks, [online] 61, pp.85–117. Available at:  
<<http://dx.doi.org/10.1016/j.neunet.2014.09.003>> [Accessed 1 Sep. 2018].

Schultz, L. and Sokolov, V., 2018. Deep Reinforcement Learning for Dynamic Urban Transportation Problems. [online] Available at: <<http://arxiv.org/abs/1806.05310>> [Accessed 4 Dec. 2018].

Seto, K.C., Güneralp, B. and Hutyrá, L.R., 2012. Global forecasts of urban expansion to 2030 and direct impacts on biodiversity and carbon pools. Proceedings of the National Academy of Sciences of the United States of America, [online] 109(40), pp.16083–8. Available at: <<http://www.ncbi.nlm.nih.gov/pubmed/22988086>> [Accessed 5 Dec. 2018].

Shabut, A.M., Lwin, K.T. and Hossain, M.A., 2016. Cyber attacks, countermeasures, and protection schemes — A state of the art survey. In: 2016 10th International Conference on Software, Knowledge, Information Management & Applications (SKIMA). [online] IEEE, pp.37–44. Available at:  
<<http://ieeexplore.ieee.org/document/7916194/>> [Accessed 8 Dec. 2018].

Shen, X.-N. and Yao, X., 2015. Mathematical modeling and multi-objective evolutionary algorithms applied to dynamic flexible job shop scheduling problems. Information Sciences, [online] 298, pp.198–224. Available at:  
<<https://www.sciencedirect.com/science/article/pii/S0020025514011177>> [Accessed 9 Sep. 2018].

Siddharth Das, 2017. CNN Architectures: LeNet, AlexNet, VGG, GoogLeNet, ResNet and more .... [online] Available at: <<https://medium.com/@sidereal/cnns-architectures-lenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5>> [Accessed 4 Dec. 2018].

Sierra, M.R. and Coello Coello, C.A., 2005. Improving PSO-Based Multi-objective Optimization Using Crowding, Mutation and  $\epsilon$ -Dominance. [online] Springer, Berlin, Heidelberg, pp.505–519. Available at: <[http://link.springer.com/10.1007/978-3-540-31880-4\\_35](http://link.springer.com/10.1007/978-3-540-31880-4_35)> [Accessed 10 Sep. 2018].

Silver, D., 2015. Lecture 2: Markov Decision Processes. UCL, Computer Science Department, Reinforcement Learning Lectures. [online] Available at:  
<[http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching\\_files/MDP.pdf](http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching_files/MDP.pdf)>.

Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T. and Hassabis, D., 2016. Mastering the game of Go with deep neural networks and tree search. [online] Available at:  
<<https://www.nature.com/articles/nature16961.pdf>> [Accessed 19 Jul. 2018].

- Slaughter, A.R., Hughes, D.A., Retief, D.C.H. and Mantel, S.K., 2017. A management-oriented water quality model for data scarce catchments. *Environmental Modelling & Software*, [online] 97(C), pp.93–111. Available at: <<https://linkinghub.elsevier.com/retrieve/pii/S1364815216309641>> [Accessed 9 Sep. 2018].
- Su, P.-H., Gašić, G., Mrkšić, N.M., Rojas-Barahona, L., Ultes, S., Vandyke, D., Wen, T.-H. and Young, S., 2016. On-line Active Reward Learning for Policy Optimisation in Spoken Dialogue Systems. [online] pp.2431–2441. Available at: <<http://www.aclweb.org/anthology/P16-1230>> [Accessed 19 Jul. 2018].
- Sutton, R.S., 1984. Temporal Credit Assignment in Reinforcement Learning. Doctoral Dissertations Available from Proquest. [online] Available at: <<https://scholarworks.umass.edu/dissertations/AAI8410337>> [Accessed 23 Sep. 2018].
- Sutton, R.S., 1988. Learning to predict by the methods of temporal differences. *Machine Learning*, [online] 3(1), pp.9–44. Available at: <<http://link.springer.com/10.1007/BF00115009>> [Accessed 8 Sep. 2018].
- Sutton, R. S., Barto, A. G., 2018. Reinforcement Learning: An Introduction, second edition. MIT Press.
- Sutton, R.S., 2018. The reward hypothesis. [online] Available at: <<http://incompleteideas.net/rlai.cs.ualberta.ca/RLAI/rewardhypothesis.html>> [Accessed 3 Aug. 2018].
- Syed, R., Suriadi, S., Adams, M., Bandara, W., Leemans, S.J.J., Ouyang, C., ter Hofstede, A.H.M., van de Weerd, I., Wynn, M.T. and Reijers, H.A., 2020. Robotic Process Automation: Contemporary themes and challenges. *Computers in Industry*, [online] 115, p.103162. Available at: <<https://linkinghub.elsevier.com/retrieve/pii/S0166361519304609>> [Accessed 17 Feb. 2020].
- Szita, I., 2012. Reinforcement Learning in Games. [online] Springer, Berlin, Heidelberg. pp.539–577. Available at: <[http://link.springer.com/10.1007/978-3-642-27645-3\\_17](http://link.springer.com/10.1007/978-3-642-27645-3_17)> [Accessed 8 Sep. 2018].
- Tajmajer, T., 2017. Modular Multi-Objective Deep Reinforcement Learning with Decision Values. [online] Available at: <<http://arxiv.org/abs/1704.06676>>.
- Takeuchi, A., Kitahashi, T. and Tanaka, K., 1975. Random environments and automata. *Information Sciences*, [online] 8(2), pp.141–154. Available at: <<https://www.sciencedirect.com/science/article/pii/0020025575900110>> [Accessed 7 Nov. 2018].
- Tanner, B. and Ca, A.U., 2009. RL-Glue: Language-Independent Software for Reinforcement-Learning Experiments Adam White. [online] *Journal of Machine Learning Research*, Available at: <<http://www.ni.uos.de/index.php?id=70>> [Accessed 5 Dec. 2018].
- Tantar, A.-A., Tantar, E. and Bouvry, P., 2011. A classification of dynamic multi-objective optimization problems. In: *Proceedings of the 13th annual conference companion on Genetic and evolutionary computation - GECCO '11*. [online] New York,

New York, USA: ACM Press.p.105. Available at:  
<<http://portal.acm.org/citation.cfm?doid=2001858.2001918>> [Accessed 6 Nov. 2018].

Thomas Simonini, 2018. Improvements in Deep Q Learning: Dueling Double DQN, Prioritized Experience Replay, and fixed.... [online] Available at:  
<<https://medium.freecodecamp.org/improvements-in-deep-q-learning-dueling-double-dqn-prioritized-experience-replay-and-fixed-58b130cc5682>> [Accessed 7 Dec. 2018].

Tian, Y., Cheng, R., Zhang, X. and Jin, Y., 2017. PlatEMO: A MATLAB Platform for Evolutionary Multi-Objective Optimization. [online] Available at:  
<<http://moeaframework.org/index.html>> [Accessed 8 Sep. 2018].

Tomas, D., Čurlin, M. and Marić, A.S., 2017. Assessing the surface water status in Pannonian ecoregion by the water quality index model. Ecological Indicators, [online] 79, pp.182–190. Available at:  
<<https://www.sciencedirect.com/science/article/pii/S1470160X17302108>> [Accessed 9 Sep. 2018].

Tooth, S. and Stephen, 2018. The geomorphology of wetlands in drylands: Resilience, nonresilience, or ...? Geomorphology, [online] 305, pp.33–48. Available at:  
<<https://linkinghub.elsevier.com/retrieve/pii/S0169555X17301319>> [Accessed 5 Dec. 2018].

Torch for RL: Introducing torch-twrl and Analytics. 2018. [online] Available at:  
<<https://analyticks.wordpress.com/2016/09/19/reinforcement-learning-for-torch-introducing-torch-twrl/>> [Accessed 5 Dec. 2018].

Torres-Jiménez, J. and Pavón, J., 2014. Applications of metaheuristics in real-life problems. Progress in Artificial Intelligence, [online] 2(4), pp.175–176. Available at:  
<<http://link.springer.com/10.1007/s13748-014-0051-8>> [Accessed 11 Nov. 2018].

Tozer, B., Mazzuchi, T. and Sarkani, S., 2017. Many-objective stochastic path finding using reinforcement learning. Expert Systems with Applications, [online] 72, pp.371–382. Available at:  
<<https://www.sciencedirect.com/science/article/pii/S0957417416305863>> [Accessed 9 Sep. 2018].

Types of Optimization Problems | NEOS. [online] 2018. Available at: <<https://neos-guide.org/optimization-tree>> [Accessed 4 Dec. 2018].

Understanding RL: The Bellman Equations. 2017. [online] joshgreaves.com. Available at: <<https://joshgreaves.com/reinforcement-learning/understanding-rl-the-bellman-equations/>> [Accessed 8 Sep. 2018].

Ursem, R.K., Ursem, R.K., Krink, T. and Filipic, B., 2002. A Numerical Simulator of a Crop-Producing Greenhouse. [online] Available at:  
<<http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.18.9724>> [Accessed 9 Sep. 2018].

Vaishya, R., Javaid, M., Khan, I.H. and Haleem, A., 2020. Artificial Intelligence (AI) applications for COVID-19 pandemic. Diabetes and Metabolic Syndrome: Clinical Research and Reviews, 14(4), pp.337–339.

- Vamplew, P., Dazeley, R., Berry, A., Issabekov, R. and Dekker, E., 2011. Empirical evaluation methods for multiobjective reinforcement learning algorithms. *Machine Learning*, 84(1–2), pp.51–80.
- Vamplew, P., Issabekov, R., Dazeley, R., Foale, C., Berry, A., Moore, T. and Creighton, D., 2017a. Steering approaches to Pareto-optimal multiobjective reinforcement learning. *Neurocomputing*, [online] 263, pp.26–38. Available at: <<https://linkinghub.elsevier.com/retrieve/pii/S0925231217311013>> [Accessed 16 Sep. 2018].
- Vamplew, P., Webb, D., Zintgraf, L.M., Roijers, D.M., Dazeley, R., Issabekov, R. and Dekker, E., 2017b. MORL-Glue: A Benchmark Suite for Multi-Objective Reinforcement Learning. [online] Available at: <<https://eportfolios.federation.edu.au/view/view.php?id=74593>> [Accessed 9 Sep. 2018].
- Veldhuizen, D.A. Van, Van Veldhuizen, D.A. and Van Veldhuizen, D.A., 1999. Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations. *EVOLUTIONARY COMPUTATION*, [online] 8, pp.125–147. Available at: <<http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.42.1823>> [Accessed 10 Oct. 2018].
- Vinyals, O., Ewalds, T., Bartunov, S., Georgiev, P., Vezhnevets, A.S., Yeo, M., Makhzani, A., Küttler, H., Agapiou, J., Schrittwieser, J., Quan, J., Gaffney, S., Petersen, S., Simonyan, K., Schaul, T., van Hasselt, H., Silver, D., Lillicrap, T., Calderone, K., Keet, P., Brunasso, A., Lawrence, D., Ekermo, A., Repp, J. and Tsing, R., 2017. StarCraft II: A New Challenge for Reinforcement Learning. [online] Available at: <<http://arxiv.org/abs/1708.04782>> [Accessed 11 Nov. 2018].
- Wang, Q., Wang, H. and Qi, Z., 2016. An application of nonlinear fuzzy analytic hierarchy process in safety evaluation of coal mine. *Safety Science*, [online] 86, pp.78–87. Available at: <<https://www.sciencedirect.com/science/article/pii/S0925753516000576>> [Accessed 19 Nov. 2018].
- Wang, W., Sebag, M., Hoi, S.C.H. and Buntine, W., 2012. Asian Conference on Machine Learning Multi-objective Monte-Carlo Tree Search. [online] Available at: <<http://proceedings.mlr.press/v25/wang12b/wang12b.pdf>> [Accessed 7 Dec. 2018].
- Wang, X., Zhang, F. and Ding, J., 2017. Evaluation of water quality based on a machine learning algorithm and water quality index for the Ebinur Lake Watershed, China. *Scientific Reports*, [online] 7(1), pp.1–18. Available at: <<http://dx.doi.org/10.1038/s41598-017-12853-y>>.
- Wang, Y. and Li, B., 2009. Investigation of memory-based multi-objective optimization evolutionary algorithm in dynamic environment. In: 2009 IEEE Congress on Evolutionary Computation. [online] IEEE, pp.630–637. Available at: <<http://ieeexplore.ieee.org/document/4983004>> [Accessed 10 Oct. 2018].
- Wang, Y. and Li, B., 2010. Multi-strategy ensemble evolutionary algorithm for dynamic multi-objective optimization. *Memetic Computing*, [online] 2(1), pp.3–24. Available at: <<http://link.springer.com/10.1007/s12293-009-0012-0>> [Accessed 9 Sep. 2018].

- Wang, Y., Zhou, J., Chen, K., Wang, Y. and Liu, L., 2017. Water quality prediction method based on LSTM neural network. 2017 12th International Conference on Intelligent Systems and Knowledge Engineering (ISKE), [online] pp.1–5. Available at: <<http://ieeexplore.ieee.org/document/8258814/>>.
- Wang, Z., Schaul, T., Hessel, M. and Lanctot, M., 2016. Dueling Network Architectures for Deep Reinforcement Learning Hado van Hasselt. [online] Available at: <<http://proceedings.mlr.press/v48/wangf16.pdf>> [Accessed 28 May 2020].
- Watkins and Dayan, 1989. Learning from delayed rewards. PhD thesis, Cambridge University. [online] Available at: <<https://ci.nii.ac.jp/naid/10000025057>> [Accessed 9 Sep. 2018].
- Wattenberg, M., Viégas, F. and Johnson, I., 2016. How to Use t-SNE Effectively. Distill, [online] 1(10), p.e2. Available at: <<http://distill.pub/2016/misread-tsne>> [Accessed 18 Nov. 2018].
- Weicker, K., 2002. Performance Measures for Dynamic Environments. [online] Springer, Berlin, Heidelberg.pp.64–73. Available at: <[http://link.springer.com/10.1007/3-540-45712-7\\_7](http://link.springer.com/10.1007/3-540-45712-7_7)> [Accessed 10 Oct. 2018].
- What is Real Reality (RR)? - Definition from Techopedia. 2018. [online] Available at: <<https://www.techopedia.com/definition/12261/real-reality-rr>> [Accessed 4 Dec. 2018].
- Wilson, A.C., Roelofs, R., Stern, M., Srebro, N. and Recht, B., 2017. The Marginal Value of Adaptive Gradient Methods in Machine Learning. [online] Available at: <<http://arxiv.org/abs/1705.08292>> [Accessed 16 Dec. 2018].
- Wilson, P.W., 2018. Dimension reduction in nonparametric models of production. European Journal of Operational Research, [online] 267(1), pp.349–367. Available at: <<https://www.sciencedirect.com/science/article/pii/S0377221717310317>> [Accessed 17 Nov. 2018].
- van Wissen, A., Gal, Y., Kamphorst, B.A. and Dignum, M.V., 2012. Human–agent teamwork in dynamic environments. Computers in Human Behavior, [online] 28(1), pp.23–33. Available at: <<http://linkinghub.elsevier.com/retrieve/pii/S0747563211001610>> [Accessed 12 Nov. 2018].
- Woo, W.L., 2020. Future trends in IM: Human-machine co-creation in the rise of AI. IEEE Instrumentation and Measurement Magazine, 23(2), pp.71–73.
- Yannakakis, G.N. and Togelius, J., 2015. A Panorama of Artificial and Computational Intelligence in Games. IEEE Transactions on Computational Intelligence and AI in Games, [online] 7(4), pp.317–335. Available at: <<http://ieeexplore.ieee.org/document/6855367/>> [Accessed 17 Feb. 2020].
- Younes, A., Calamai, P. and Basir, O., 2005. Generalized benchmark generation for dynamic combinatorial problems. In: Proceedings of the 2005 workshops on Genetic and evolutionary computation - GECCO '05. [online] New York, New York, USA: ACM Press.p.25. Available at: <<http://portal.acm.org/citation.cfm?doid=1102256.1102262>> [Accessed 4 Nov. 2018].

- Young, T., Hazarika, D., Poria, S. and Cambria, E., 2018. Recent Trends in Deep Learning Based Natural Language Processing [Review Article]. *IEEE Computational Intelligence Magazine*, [online] 13(3), pp.55–75. Available at: <<https://ieeexplore.ieee.org/document/8416973/>> [Accessed 4 Dec. 2018].
- Younis, M., 2018. Internet of everything and everybody: Architecture and service virtualization. *Computer Communications*, [online] 131, pp.66–72. Available at: <<https://www.sciencedirect.com/science/article/pii/S0140366418303116>> [Accessed 4 Dec. 2018].
- Zanzotto, F.M., 2019. Viewpoint: Human-in-the-loop artificial intelligence. In: *CEUR Workshop Proceedings*. CEUR-WS.pp.84–94.
- Zaroliagis, C. and Christos, 2005. Recent Advances in Multiobjective Optimization. In: *Proceedings of the Third international conference on Stochastic Algorithms: foundations and applications*. [online] Springer-Verlag.pp.45–47. Available at: <[http://link.springer.com/10.1007/11571155\\_5](http://link.springer.com/10.1007/11571155_5)> [Accessed 19 Nov. 2018].
- Zeng, S., Zeng, L., Dong, X. and Chen, J., 2013. Polycyclic aromatic hydrocarbons in river sediments from the western and southern catchments of the Bohai Sea, China: toxicity assessment and source identification. *Environmental Monitoring and Assessment*, [online] 185(5), pp.4291–4303. Available at: <<http://www.ncbi.nlm.nih.gov/pubmed/22983614>> [Accessed 5 Dec. 2018].
- Zhang, Y., Zhang, L., Neoh, S.C., Mistry, K. and Hossain, M.A., 2015. Intelligent affect regression for bodily expressions using hybrid particle swarm optimization and adaptive ensembles. *Expert Systems with Applications*, [online] 42(22), pp.8678–8697. Available at: <<https://www.sciencedirect.com/science/article/pii/S0957417415004820>> [Accessed 8 Dec. 2018].
- Zhang, Z. and Qian, S., 2011. Artificial immune system in dynamic environments solving time-varying non-linear constrained multi-objective problems. *Soft Computing*, [online] 15(7), pp.1333–1349. Available at: <<http://link.springer.com/10.1007/s00500-010-0674-z>> [Accessed 10 Oct. 2018].
- Zheng, B., 2007. A New Dynamic Multi-objective Optimization Evolutionary Algorithm. In: *Third International Conference on Natural Computation (ICNC 2007) Vol V*. [online] IEEE.pp.565–570. Available at: <<http://ieeexplore.ieee.org/document/4344903/>> [Accessed 10 Oct. 2018].
- Zheng, K., Li, H., Qiu, R.C. and Gong, S., 2012. Multi-objective reinforcement learning based routing in cognitive radio networks: Walking in a random maze. In: *2012 International Conference on Computing, Networking and Communications (ICNC)*. [online] IEEE.pp.359–363. Available at: <<http://ieeexplore.ieee.org/document/6167444/>> [Accessed 9 Sep. 2018].
- Zhou, A., Zhao, H., Zhang, H. and Zhang, G., 2019. Pareto optimal set approximation by models: A linear case. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Springer Verlag.pp.451–462.
- Zhou, H.-B., Chen, T.-B., Gao, D., Zheng, G.-D., Chen, J., Pan, T.-H., Liu, H.-T. and Gu, R.-Y., 2014. Simulation of water removal process and optimization of aeration

strategy in sewage sludge composting. *Bioresource technology*, [online] 171, pp.452–60. Available at: <<http://www.ncbi.nlm.nih.gov/pubmed/25233360>> [Accessed 8 Sep. 2018].

Zitzler, E. and Thiele, L., 1999. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation*, [online] 3(4), pp.257–271. Available at: <<http://ieeexplore.ieee.org/document/797969/>> [Accessed 10 Oct. 2018].



## Appendices

### Appendix A: Treasure values and the Pareto frontiers

In this appendix, treasure distributions have been shown in Table A.1. Identified treasure values (i.e. Pareto frontier) for the dynamic DST (silver and gold) have been shown in Table A.2. Two scenarios are described in the dynamic DST (attack by enemy) environment with the identified treasure values along with the time cost.

To get more details, click the following video link:

(<https://www.dropbox.com/sh/joku75dhwckjhbu/AAAdPEr2lOyZjdMe0Q-fI3eTa?dl=0>) which has been created while traversing the agent in different scenarios.

Table A. 1: DST testbed treasure values

Treasure values		
DST (silver and gold)	Silver	1,2,3,5,8,16,24,50,74,124
	Gold	100,925,1231,1442,1525,1597,1797,1829,1889,1900
DST (attack by enemy)	1,2,3,5,8,16,24,50,74,124	

Table A. 2: Treasure values for the Pareto Frontier (Silver and Gold)

Treasure Types	Identified Treasures
Silver	1, 2, 3, 5, 50, 74, 124
Gold	1597, 1797, 1829

Table A. 3: Treasure values for the Pareto Frontier (attack by enemy- scenario 1)

Identified Treasures	Time Cost	Health Meter
1	1	10
2	3	10
0	4	8
3	5	10
5	7	10
8	8	8
16	9	10
24	13	10
50	14	10
74	17	10
124	19	2

Table A. 4: Treasure values for the Pareto Frontier (attack by enemy- scenario 2)

Identified Treasures	Time Cost	Health Meter
1	1	10
2	3	10
3	5	10
5	7	10
8	8	10
16	9	10
24	13	10
50	14	8
74	17	10
124	19	8

## Appendix B: Sample WQR dataset

A sample dataset has mentioned in Table B.1. Here, the stations of the zone 1 and 2 have been mentioned. The language of the raw dataset is in Portuguese.

Table B. 1: Sample raw dataset for the WQR

UGRHI														
	Corpo Hídrico	Ponto	Jan	Fev	Mar	Abr	Mai	Jun	Jul	Ago	Set	Out	Nov	Dez
1	Rio da Prata	PRAT02400		61		53		60		58		45		51
		SAGU02050		57		53		60		52		53		55
	Rio Sapucaí Guaçu	SAGU02250		71		60		63		65		63		59
	Rio Sapucaí-Mirim	SAMI02200		59		55		62		61		55		59
	Braço do Rio Paraibuna	IUNA00950		91		85		89		89		87		88
	Braço do Rio Paraitinga	INGA00850		89		83		87		88		89		90
	Cór. do Pontilhão ou Barrinha	PONT04950		15		17		26		13		14		15
2	Reservatório do Jaguari - UGRHI 02	JAGJ00200		81		80		83		81		64		80
		JAGJ00900		87		83		86		89		89		86
	Reservatório Santa Branca	SANT00100		84		82		79		84		82		91
	Ribeirão da Água Limpa	ALIM02950		48		44		31		41		31		37
	Rio Buquira	BUKI02950		45		57		56		50		49		51
	Rio Guaratingueta	GUAT02800		73		65		75		69		72		73

The following Table B.2 shows the threshold values of the IQA, IET and IVA.

Table B. 2: The threshold level to determine the resilience of IQA, IVA and IET

Index	Categories					
IQA	High quality	Good quality	Average quality		Poor quality	Very poor quality
	$79 < IQA \leq 100$	$51 < IQA \leq 79$	$36 < IQA \leq 51$		$19 < IQA \leq 36$	$IQA \leq 19$
IVA	$IVA \leq 2.5$	$2.6 < IVA \leq 3.3$	$3.4 < IVA \leq 4.5$		$4.6 < IVA \leq 6.7$	$6.8 \leq IVA$
IET	High oligotrophic	Oligotrophic	Mesotrophic	Eutrophic	Highly eutrophic	Very highly eutrophic
	$IET \leq 47$	$47 < IET \leq 52$	$52 < IET \leq 59$	$59 < IET \leq 63$	$63 < IET \leq 67$	$67 < IET$

## Appendix C: Keras implementation

Keras Implementation for SGD:

```
sgd = SGD(lr=0.1, decay=1e-6, momentum=0.9, nesterov=True)
```

```
model.compile(loss='mean_squared_error', optimizer=sgd)
```

Keras Implementation for Adam:

```
keras.optimizers.Adam(lr=0.001, beta_1=0.9, beta_2=0.999, epsilon=None,
decay=0.0, amsgrad=False)
```

```
model.compile(loss='mean_squared_error', optimizer=adam)
```

## Appendix D: Visualisation of the deep layers

In this section, a detailed visualisation of the deep layers has been illustrated which is generated by the TensorBoard.

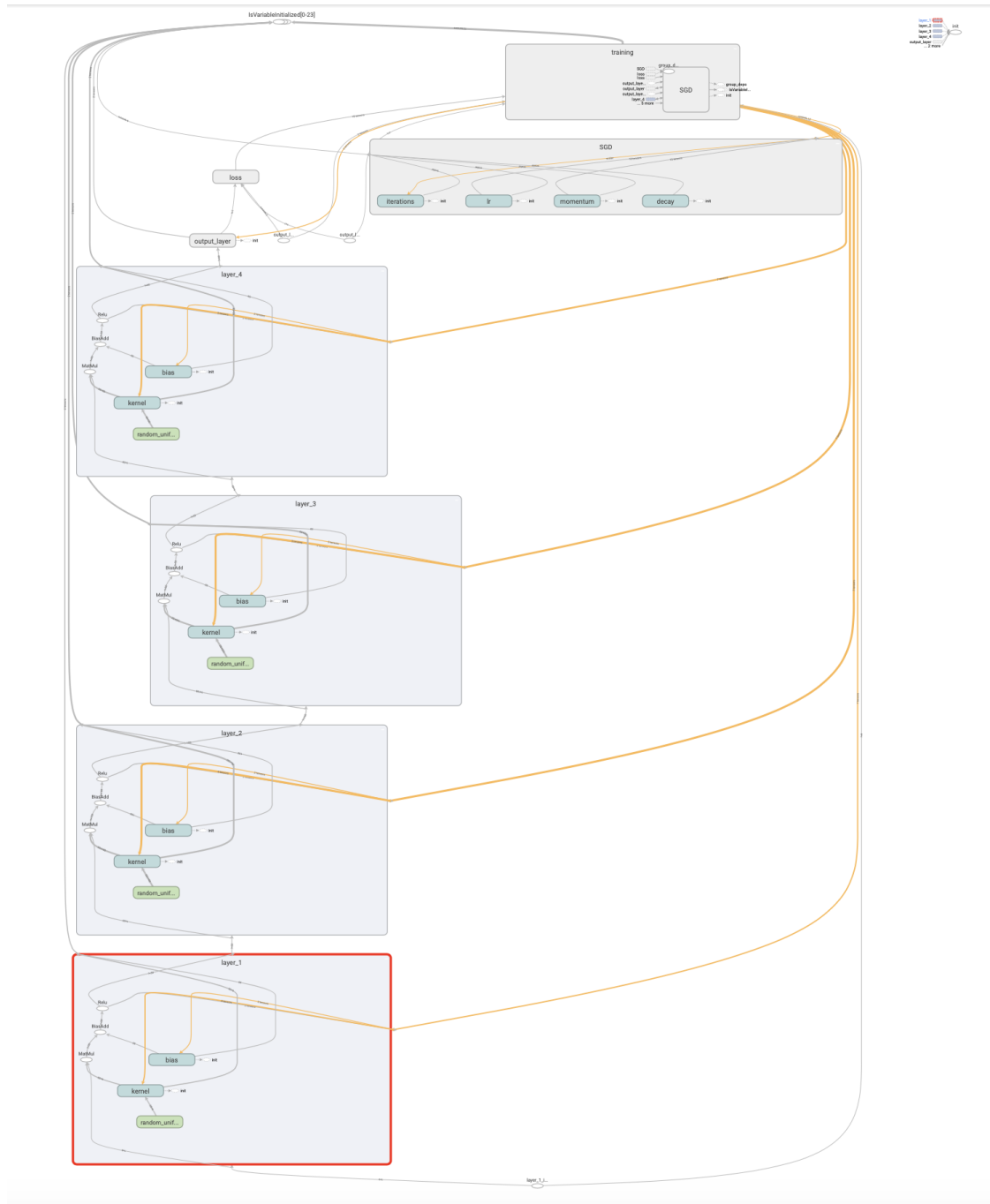


Figure D. 1: Bird's eye view of the deep layer network

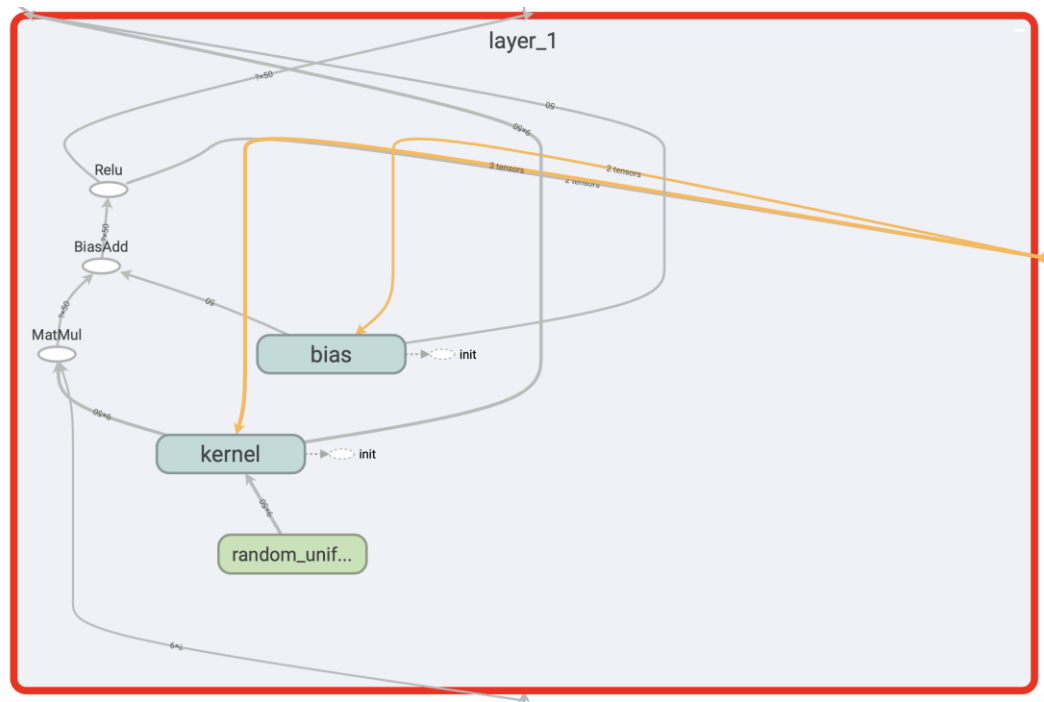


Figure D. 2: Visualisation of deep layer 1

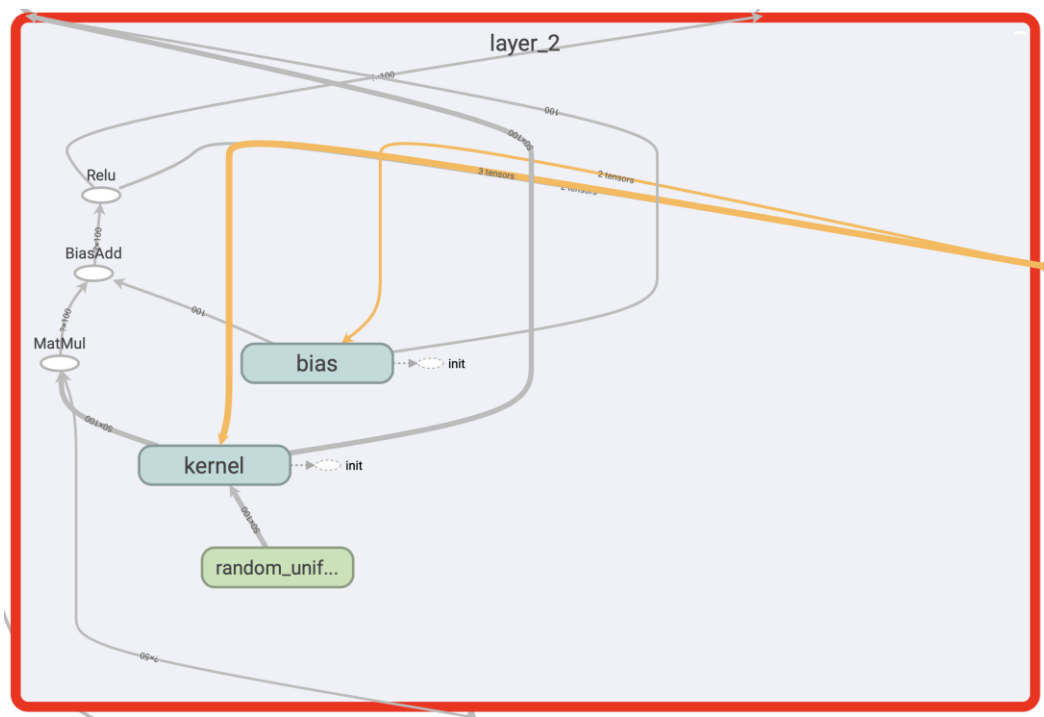


Figure D. 3: Visualisation of deep layer 2

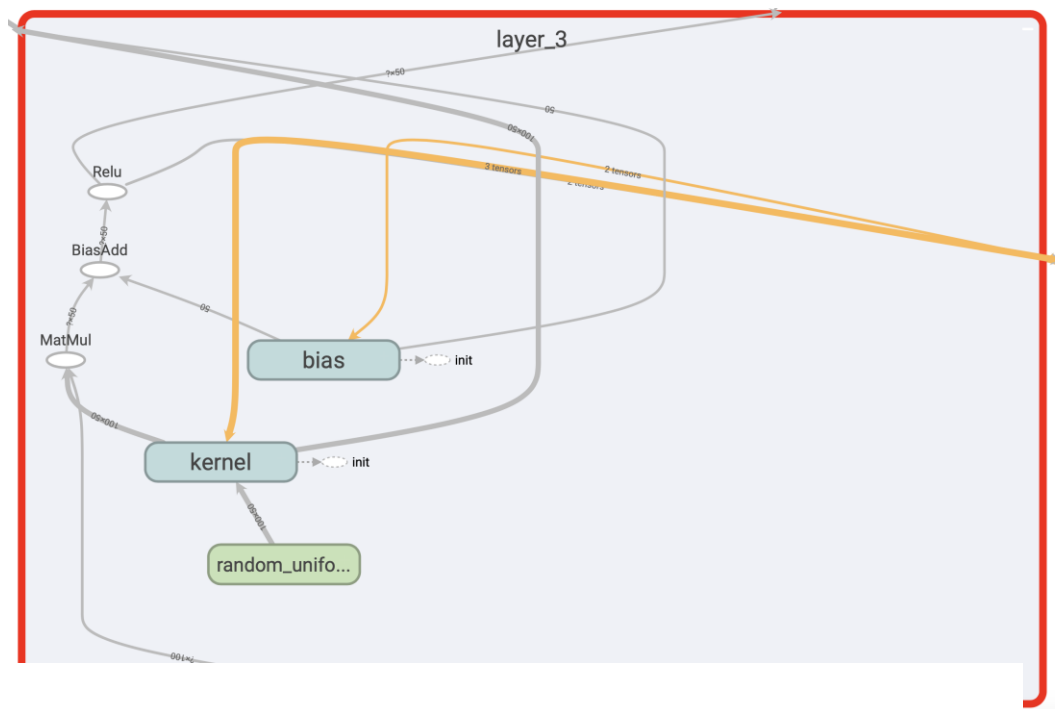


Figure D. 4: Visualisation of deep layer 3

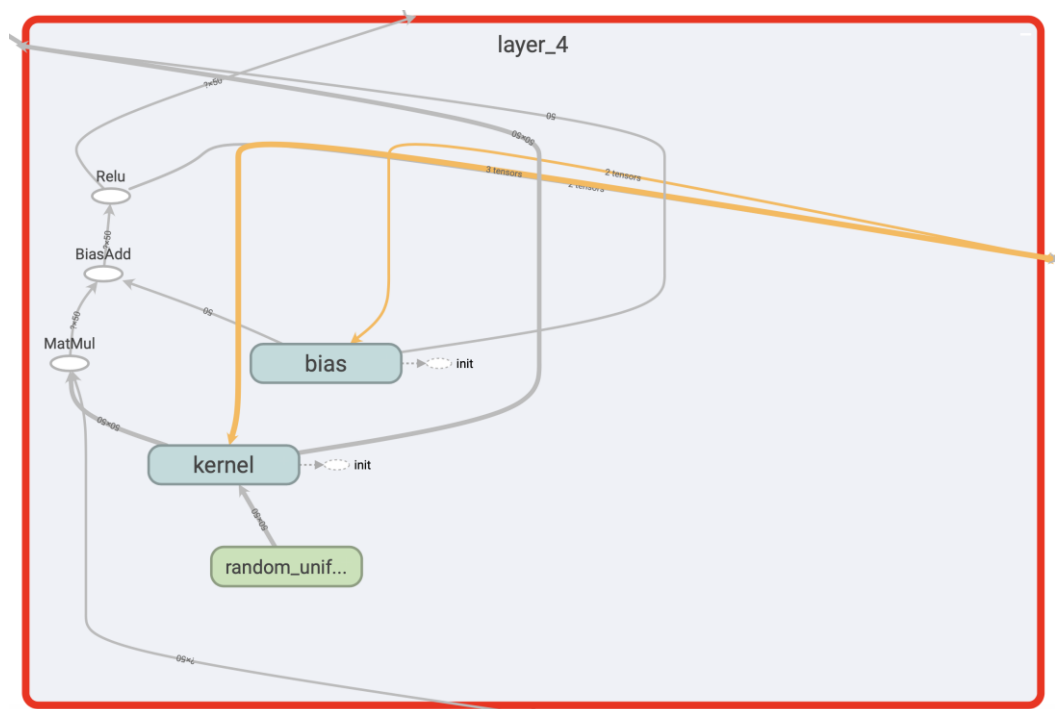


Figure D. 5: Visualisation of deep layer 4

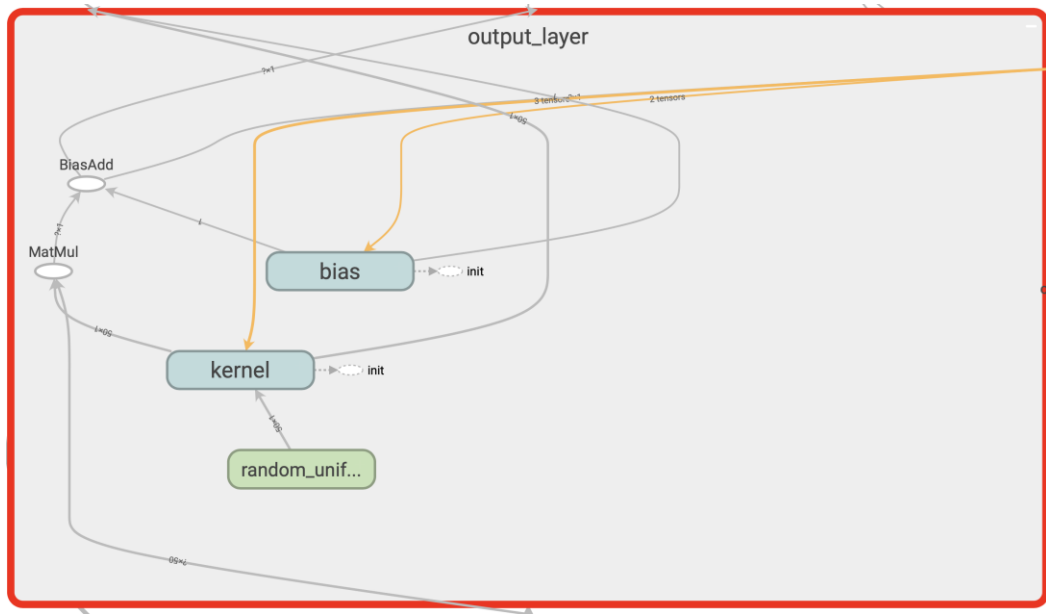


Figure D. 6: Visualisation of the output layer

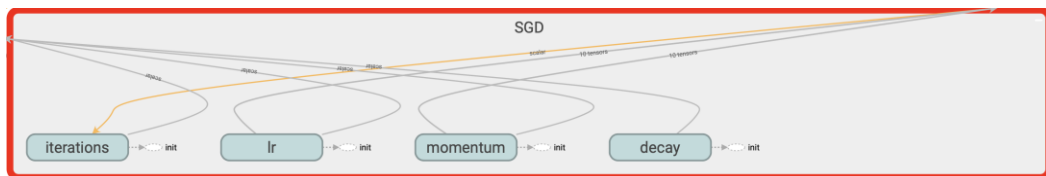


Figure D. 7: Visualisation of the stochastic gradient descent (SGD) layer

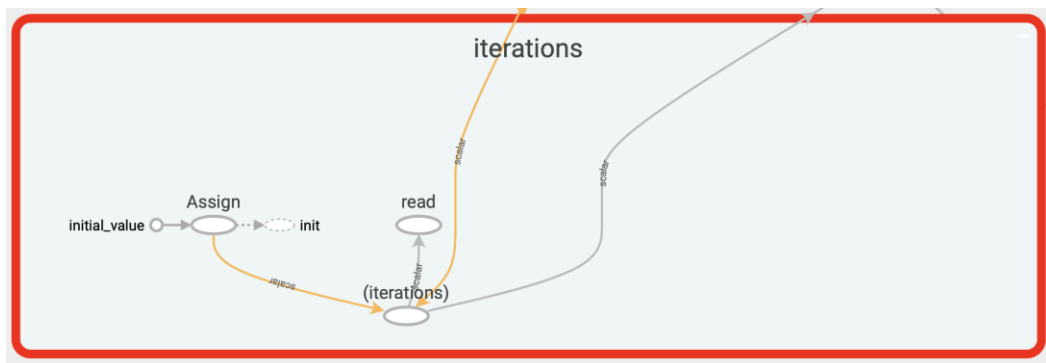


Figure D. 8: Visualisation of the iterations

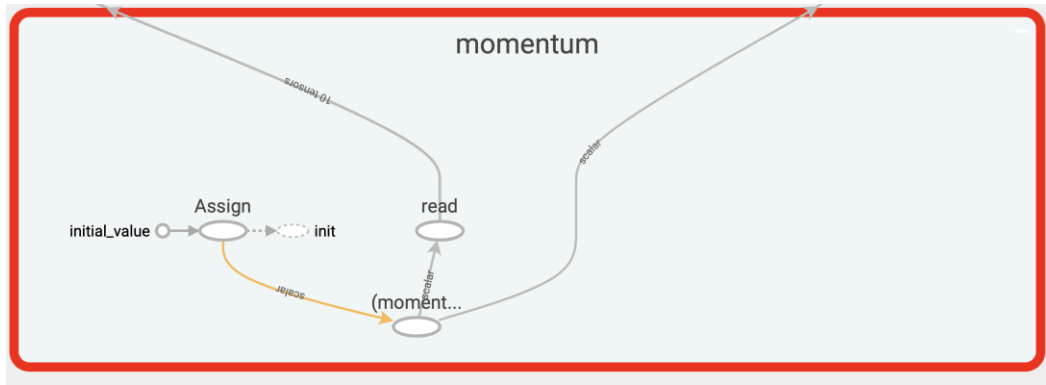


Figure D. 9: Visualisation of the momentum layer

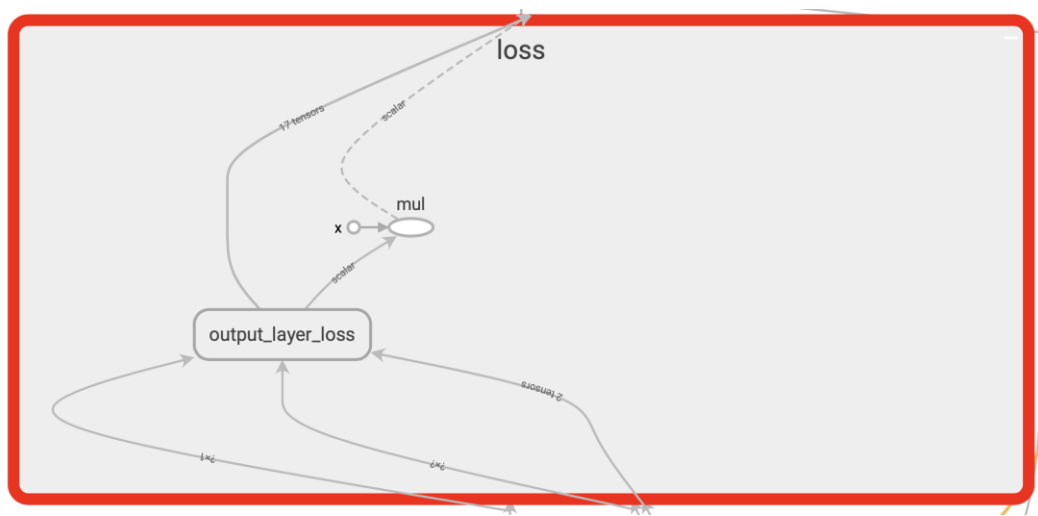


Figure D. 10: Visualisation of the loss layer

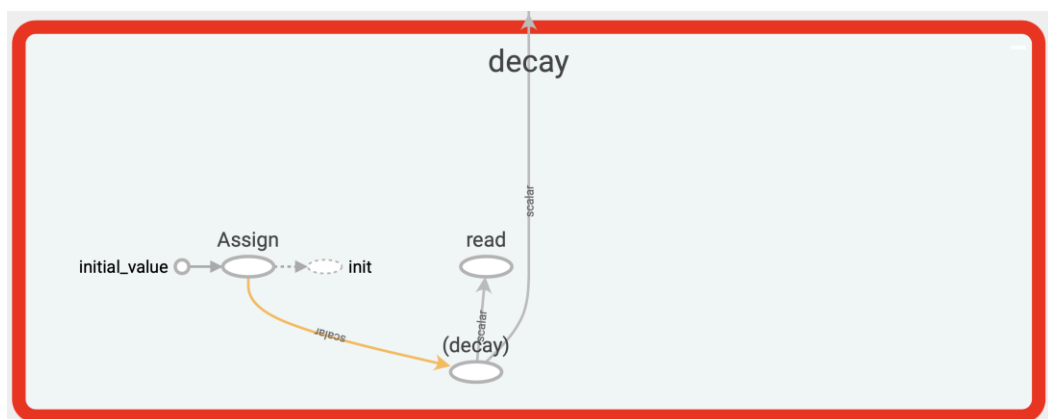


Figure D. 11: Visualisation of the decay layer



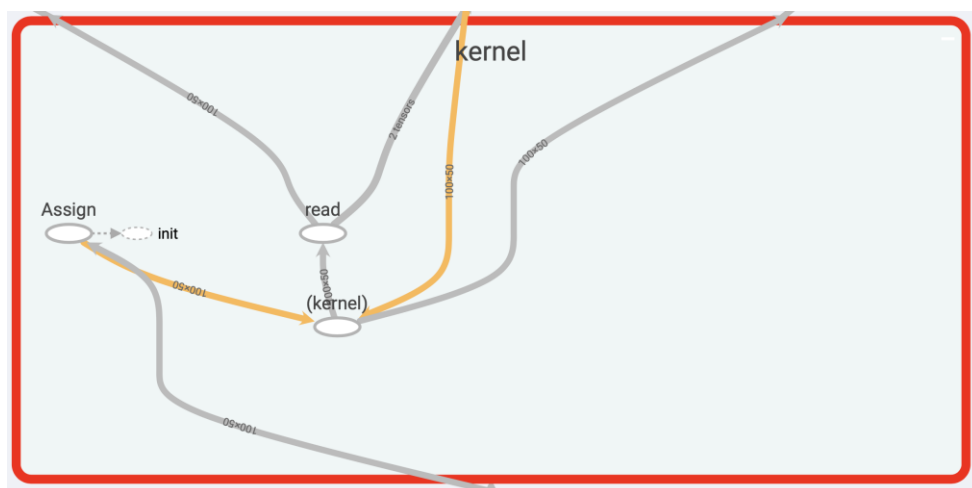


Figure D. 12: Visualisation of the kernel layer

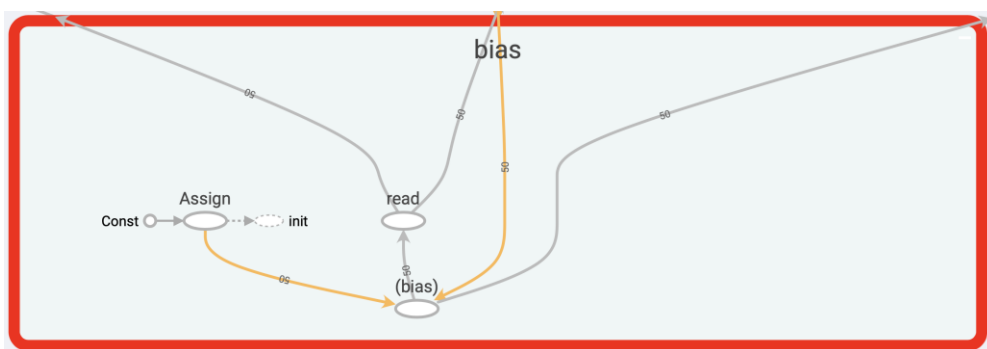


Figure D. 13: Visualisation of the bias layer

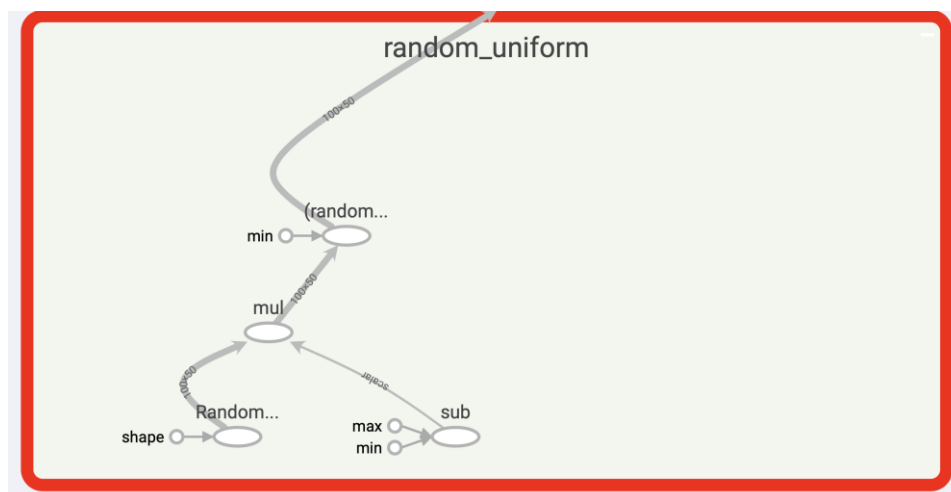


Figure D. 14: Visualisation of the random\_uniform layer

## Appendix E: Weight-bias distributions

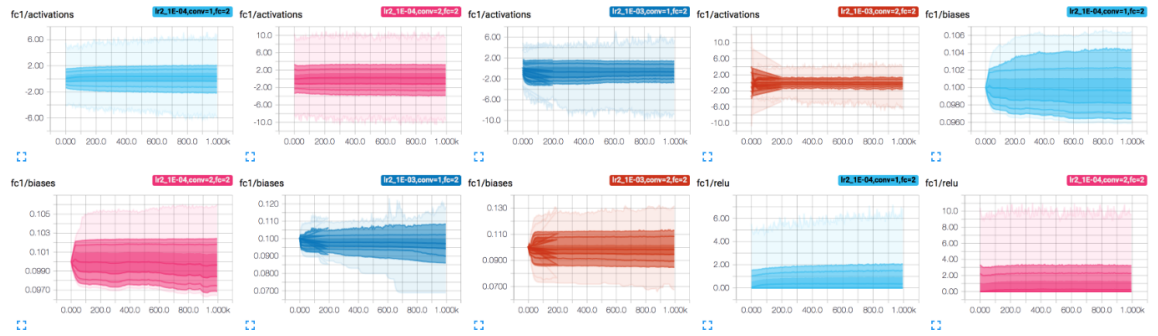


Figure E. 1: Weight-bias distribution for dynamic DST (silver and gold)



Figure E. 2: Weight-bias distribution for dynamic DST (attack by enemy)

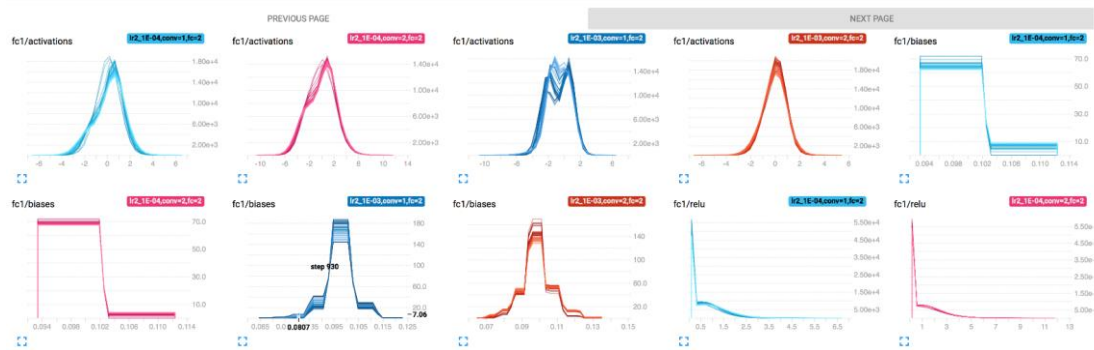


Figure E. 3: Weight-bias distribution for WQR environment

## Appendix F: Expert system and identified vulnerable zones

The Expert System (ES) is developed for identifying the water quality and predicting the vulnerable zones as a standalone system and can be accessed with the MATLAB extension. The external user of the proposed ES can be accessed through a web browser over the Internet. As a result, the system can be used in PC and mobile device. However, for the mobile device, MATLAB Mobile needs to be installed. The following link is the Git repository (i.e. source code) for the MATLAB code.

Link: <https://gitlab.com/mahmudul.05/wqn-brazil>

Figure F.1 shows the login screen of the ES to predict vulnerable zones using water quality resilience.



Figure F. 1: Login window of the ES

Figure F.2 demonstrates the resilience prediction based on the IQA, IET and IVA associated with the overall impact for the corresponding stations out of 22 zones.

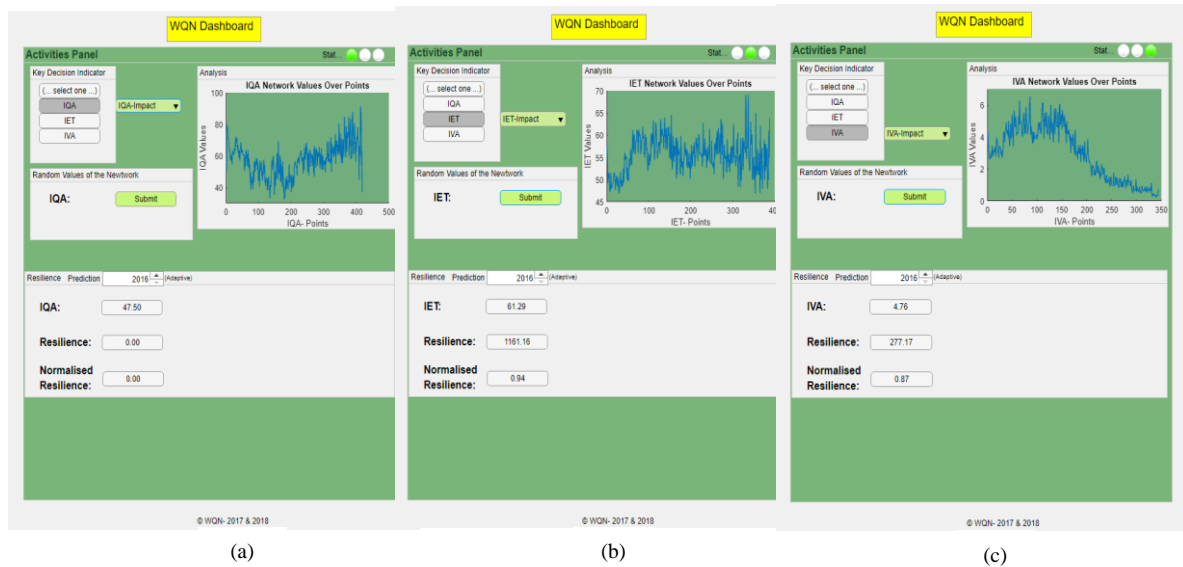


Figure F. 2: ES to predict water quality resilience based on (a) IQA, (b) IET and (c) IVA

Figure F.3 shows the outcome of the ES such as the most vulnerable zones based on a particular zone. Here, four vulnerable stations are identified for zone 5.

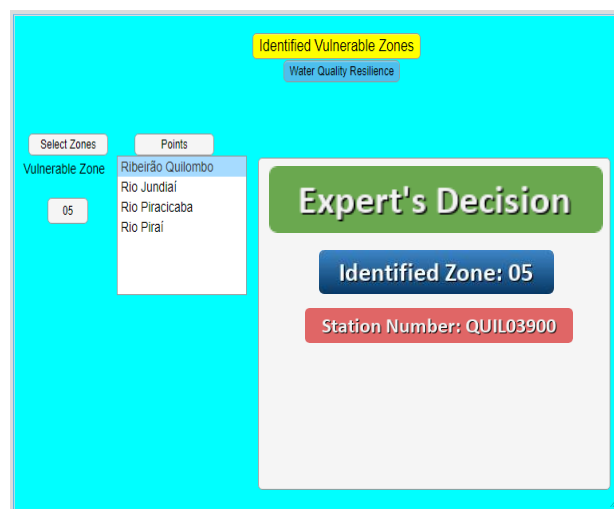


Figure F. 3: Identified vulnerable stations of Zone 5 by the ES

Identified vulnerable zones based on IQA, IVA and IET have been shown in Figure F.4, Figure F.5 and Figure F.6 respectively.

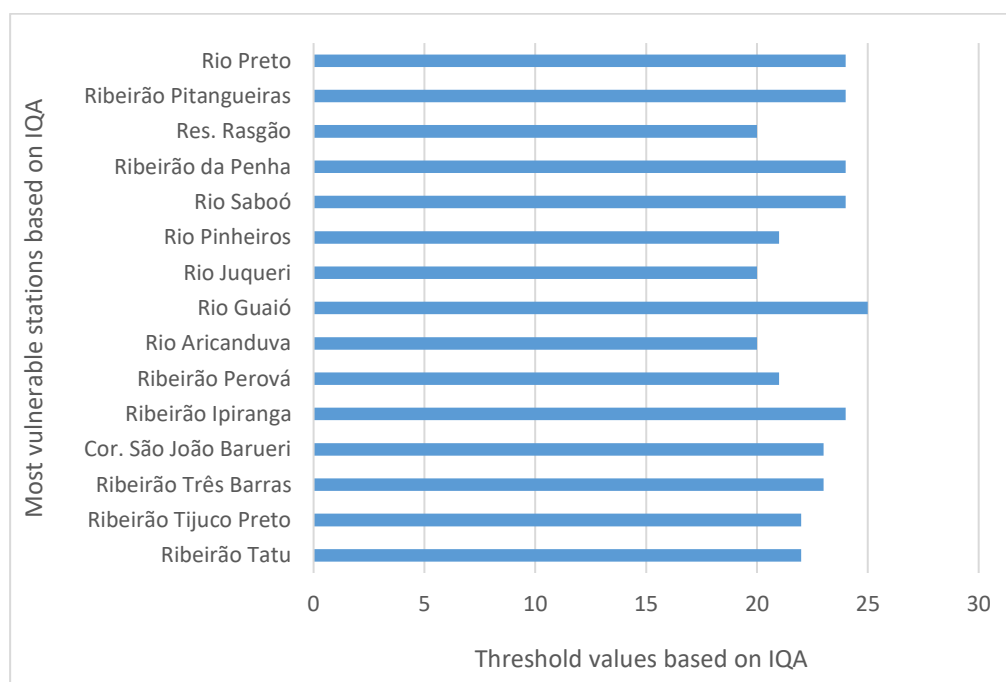


Figure F. 4: Identified the most vulnerable stations based on IQA

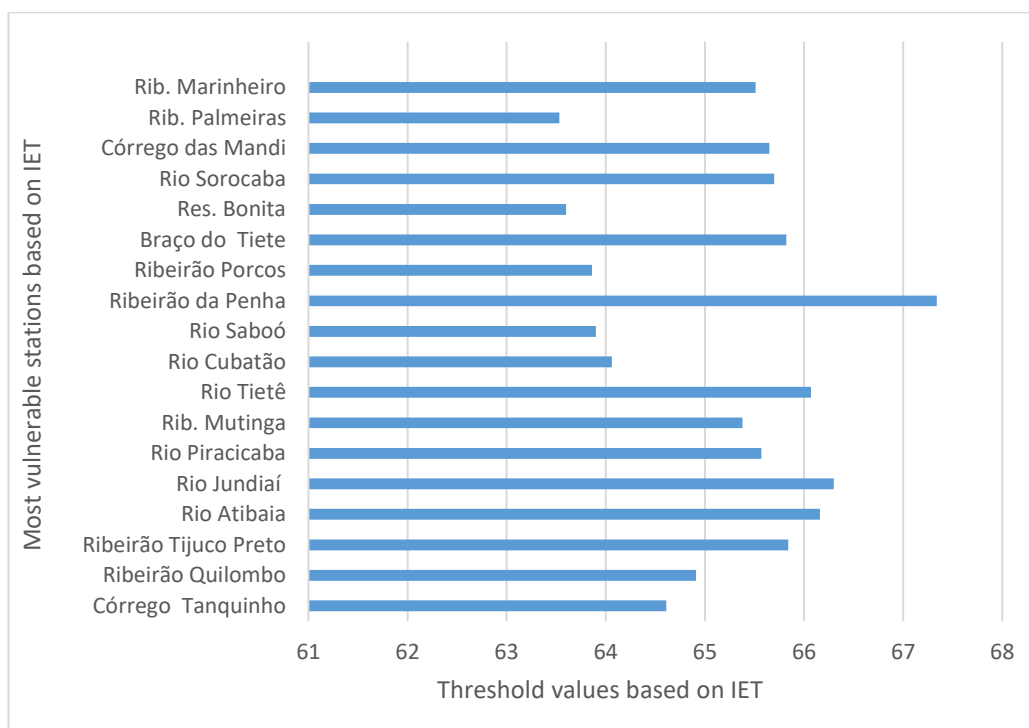


Figure F. 5: Identified the most vulnerable stations based on IET

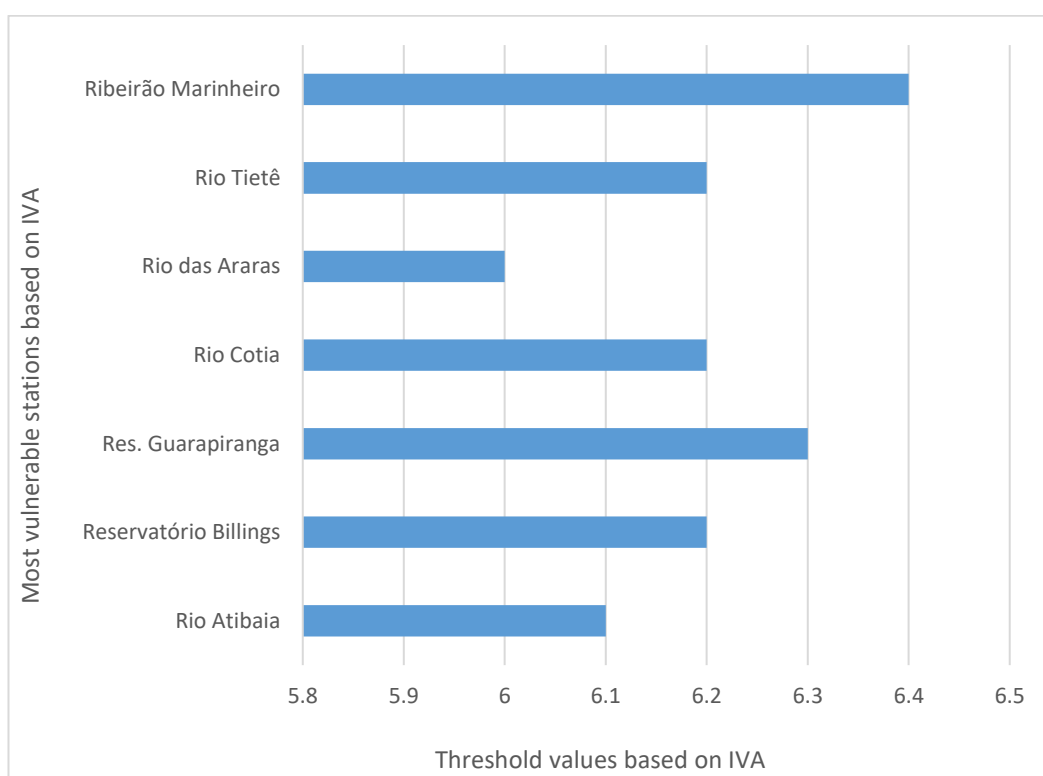


Figure F. 6: Identified the most vulnerable stations based on IVA

Table F.1 shows the most critical zones and their physical location based on latitude and longitude.

Table F. 1: Physical location of the most vulnerable zones

Name	Zone	Latitude	Longitude
Ribeirão Quilombo	5	-15.58568	-56.10695
Reservatório do Guarapiranga	6	-21.97582	-48.25019
Ribeirão do Marinho	15	-27.57763	-48.59243