Diss. ETH No. 29156

PRESCRIPTIVE MAINTENANCE AND OPERATION WITH DEEP REINFORCEMENT LEARNING

A dissertation submitted to attain the degree of

Doctor of Sciences of ETH Zurich
(Dr. sc. ETH Zurich)

presented by

Yuan Tian

MSc ME, TU Delft

born on 03.01.1995

citizen of China

accepted on the recommendation of

Prof. Dr. Olga Fink, [examiner]
Prof. Dr. David Coit, [co-examiner]
Prof. Dr. Zhiwu Huang, [co-examiner]
Prof. Dr. Giovanni Sansavini, [co-examiner]

2023

Abstract

Although predictive maintenance has improved the availability of industrial systems by predicting the remaining useful life and scheduling maintenance actions in a timely manner, it only provides information on the necessity of performing or deferring maintenance on assets. However, for mission-critical applications, taking action and providing information to delay equipment failure and prolong their working cycle or remaining useful life can be crucial to the deployability of devices. Prescriptive maintenance goes beyond the prediction of the end of life and aims to prescribe optimal operation with respect to the health condition and the expected usage profile. Prescribed operations can help proactively manage the performance, reliability, and availability of a system. To perform prescriptive maintenance, the dynamical degradation process and the subsequent impact of the current decision on system health are required to be integrated into the control actions or respective decisions. However, the fact that the explicit degradation features may not be directly available or cannot be inferred easily in real-world scenarios limits the applications of model-based optimization approaches for prescriptive maintenance. Moreover, model-based optimization approaches suffer from high computational costs, which also precludes their application to more complex systems, distributed systems, or immediate-level operation optimizations. To address the computational challenge and the lack of explicit health information, one promising potential solution for such tasks is reinforcement learning, which has recently shown extraordinary capability on a wide range of end-to-end control tasks. Motivated by the achievements of reinforcement learning in other fields, in this dissertation, we aim to develop a reinforcement learning-based prescriptive maintenance framework that integrates degradation awareness into decision-making. The proposed framework can perform in real-time and can improve the system's performance and prolong its remaining useful life. Within the framework, four modules are proposed and detailed below.

First of all, in order to compensate for the deviation between sensor measurements and outputs of physics-based performance models due to the degradation process on the real system, we develop a real-time model calibration module, which can infer the underlying degradation parameters and thereby calibrate the simulation environments. This module plays an essential role in the proposed framework to guarantee well-calibrated training environments for developing prescriptive maintenance applications. Moreover, the proposed module can also provide a better awareness of the degradation process.

Secondly, we develop a real-time load allocation module for multi-battery systems that, for the first time, can proactively distribute load demand to each battery based on only raw sensor measurements. This module can effectively prolong the working cycle and remaining useful life of the deployed multi-battery systems. This module has an essential impact on the reliability and deployability of multi-battery systems for safety-critical applications. As a methodological contribution, the Dirichlet policy is proposed for the first time to improve the effectiveness and learning stability for continuous action space allocation tasks under simplex constraint. The Dirichlet policy can be combined with any other reinforcement learning algorithms to improve the performance of solving allocation problems.

Thirdly, in addition to the independent decisions, in some cases (e.g., distributed systems), the decisions can implicitly or explicitly influence other systems, which are usually modeled as multi-agent systems. Since the centralized optimization methods are less flexible and suffer from high computational costs, we propose a novel decentralized multi-agent reinforcement learning method to address these types of problems. The proposed method follows the de-

centralized execution mechanism, which is more flexible and computationally more efficient compared to centralized execution for real-time decision-making. Moreover, we propose a general opponent model that allows the agent to tackle mixed cooperative-competitive scenarios, which has been less studied but is widely relevant in real-world applications. This module serves as a complement to the previous modules when the structural and operational dependencies within each machine or system must be considered.

Lastly, we propose an effective and efficient optimization module, especially for process optimization or system reconfiguration problems in the industry. In such tasks, the analytical forms of some or all of the functions and constraints are unknown or unexploitable, which makes classical gradient-based approaches inapplicable. This work is the final piece of the proposed immediate-level prescriptive maintenance framework, which provides an efficient solution for such optimization tasks. With the novel problem modeling and the introduced adaptive searching strategy, the proposed module can prescribe reliable and reproducible recommendations with limited data, which is in line with the industrial need.

In this thesis, we demonstrate the effectiveness and efficiency of the proposed framework and corresponding modules on challenging industrial use cases or reinforcement learning benchmarks. The proposed prescriptive maintenance framework, based on reinforcement learning, shows a promising potential to improve the performance, reliability, and lifetime of industrial systems while reducing maintenance costs.

Zusammenfassung

Prädiktive Instandhaltung hat zwar die Verfügbarkeit industrieller Systeme durch die Vorhersage der Restnutzungsdauer und die rechtzeitige Planung von Instandhaltungsmassnahmen verbessert, liefert aber nur Informationen über die Notwendigkeit der Durchführung oder des Aufschubs von Instandhaltungsarbeiten an den Anlagen. Bei unternehmenskritischen Anwendungen kann es jedoch für die Einsatzfähigkeit von Geräten entscheidend sein, Präventivmassnahmen zu ergreifen und Informationen bereitzustellen, um den Ausfall von Geräten zu verzögern und ihren Arbeitszyklus oder ihre Restnutzungsdauer zu verlängern. Die präskriptive Instandhaltung geht über die Vorhersage der Restnutzungsdauer hinaus und zielt darauf ab, den optimalen Betrieb im Hinblick auf den Gesundheitszustand und das erwartete Nutzungsprofil vorzuschreiben. Vorgeschriebene Abläufe können helfen, die Leistungsfähigkeit, Zuverlässigkeit und Verfügbarkeit eines Systems proaktiv zu beeinflussen. Um präskriptive Instandhaltung durchzuführen, müssen der dynamische Degradationsprozess und die resultierenden Auswirkungen der aktuellen Entscheidung auf den Systemzustand in die Steuerungsmassnahmen oder entsprechenden Entscheidungen integriert werden. Die Tatsache, dass die expliziten Degradationsmerkmale nicht direkt verfügbar sind oder in realen Szenarien nicht einfach inferiert werden können, schränkt jedoch die Anwendung von modellbasierten Optimierungsansätzen für präskriptive Instandhaltung ein. Darüber hinaus ist die Berechnung von modellbasierten Optimierungsansätzen mit hohen Rechenkosten verbunden, was ihre Anwendung auf komplexere Systeme, verteilte Systeme oder Betriebsoptimierungen in Echtzeit ebenfalls ausschliesst. Eine vielversprechende Lösung für solche Aufgaben ist das Reinforcement Learning, das in jüngster Zeit bei einer Vielzahl von End-to-End-Regelungsaufgaben im Bezug auf die Rechenzeit und dem Mangel an expliziten Gesundheitsinformationen aussergewöhnliche Performance aufzeigte. Motiviert durch die Errungenschaften von Reinforcement Learning in anderen Bereichen, zielen wir in dieser Dissertation darauf ab, ein auf Reinforcement Learning basierendes Framework für präskriptive Instandhaltung zu entwickeln, das die Information über den Degradationszustand in die Entscheidungsfindung einbezieht. Das vorgeschlagene System ist in der Lage, in Echtzeit zu arbeiten, die Systemleistung zu verbessern und die verbleibende Nutzungsdauer zu verlängern. Innerhalb des Frameworks werden vier Module vorgeschlagen, die im Folgenden näher erläutert werden.

Um die Abweichungen zwischen den Sensormessungen und den Ergebnissen der physikalischen Leistungsmodelle, die aufgrund des Degradationsprozesses eintreten, im realen System zu kompensieren, entwickeln wir zunächst ein Echtzeit-Modellkalibrierungsmodul, das die zugrundeliegenden Degradationsparameter inferiert und damit die Simulationsumgebungen kalibrieren kann. Dieses Modul spielt eine wesentliche Rolle in dem vorgeschlagenen Framework, um gut kalibrierte Trainingsumgebungen für die Entwicklung von präskriptiven Instandhaltungsanwendungen zu gewährleisten. Darüber hinaus kann das vorgeschlagene Modul auch eine bessere Kenntniss über den Degradationsprozess schaffen.

Zweitens entwickeln wir ein Echtzeit-Lastzuteilungsmodul für Systeme mit mehreren Batterien, das erstmals in der Lage ist, den Lastbedarf proaktiv auf jede Batterie zu verteilen, und zwar ausschliesslich auf der Grundlage von Rohsensormessungen. Dieses Modul kann den Arbeitszyklus und die verbleibende Nutzungsdauer der eingesetzten Multibatteriesysteme effektiv verlängern und hat daher einen wesentlichen Einfluss auf die Zuverlässigkeit und Einsatzfähigkeit von Mehrbatteriesystemen für sicherheitskritische Anwendungen. Als methodischer Beitrag wird zum ersten Mal die Dirichlet-Policy vorgeschlagen, um die Effektivität und Lernstabilität für kontinuierliche Aktionsraum-Allokationsaufgaben unter der Simplex-

Nebenbedingung zu verbessern. Die Dirichlet-Policy kann mit allen bestehenden Reinforcement Learning Algorithmen kombiniert werden, um die Leistungsfähigkeit bei der Lösung von Allokationsproblemen zu verbessern.

Drittens können Regelungsentscheidungen in einigen Fällen (z. B. bei verteilten Systemen) implizit oder explizit andere Systeme beeinflussen, die in der Regel als Multiagentensysteme modelliert werden. Da die zentralisierten Optimierungsmethoden weniger flexibel sind und mit hohen Rechenkosten verbunden sind, schlagen wir eine neuartige dezentrale Multi-Agenten-Reinforcement Learning Methode vor, um diese Art von Problemen anzugehen. Die vorgeschlagene Methode folgt einer dezentralisierten Berechnungsstrategie, die flexibler und rechnerisch effizienter ist als die zentralisierte Berechnung von Echtzeitentscheidungen. Darüber hinaus schlagen wir ein allgemeines Oponent Model vor, das es dem Agenten ermöglicht, gemischte kooperativ-kompetitive Szenarien zu bewältigen, was bisher weniger untersucht wurde, aber in realen Anwendungen von grosser Bedeutung ist. Dieses Modul dient als Ergänzung zum vorherigen Modul, wenn die strukturellen und betrieblichen Abhängigkeiten zwischen den einzelnen Maschinen oder Systemen berücksichtigt werden müssen.

Schliesslich schlagen wir ein effektives und effizientes Optimierungsmodul vor, insbesondere für Prozessoptimierungs- oder Systemrekonfigurationsprobleme in der Industrie. Bei solchen Aufgaben sind die analytischen Formen einiger oder aller Funktionen und Nebenbedingungen unbekannt oder unverwertbar, so dass klassische gradientenbasierte Ansätze nicht anwendbar sind. Diese Arbeit ist der letzte Baustein des vorgeschlagenen Frameworks für die präskriptive Echtzeitinstandhaltung, die eine effiziente Lösung für solche Optimierungsaufgaben bietet. Durch die neuartige Problemmodellierung und die eingeführte adaptive Suchstrategie kann das vorgeschlagene Modul zuverlässige und reproduzierbare Empfehlungen mit sehr wenig Daten machen, was dem industriellen Bedarf entspricht.

In dieser Arbeit demonstrieren wir die Effektivität und Effizienz des vorgeschlagenen Frameworks und der zugehörigen Module anhand von anspruchsvollen industriellen Anwendungsfällen oder Reinforcement Learning Benchmarks. Das vorgeschlagene präskriptive Instandhaltungskonzept, das auf Reinforcement Learning basiert, zeigt ein vielversprechendes Potenzial zur Verbesserung der Leistungsfähigkeit, Zuverlässigkeit und Langlebigkeit industrieller Systeme bei gleichzeitiger Reduzierung der Instandhaltungskosten.

Contents

Acknowledgments

During my 3.5-year doctoral journey at ETH Zurich, I gained invaluable experience and had the privilege of meeting many remarkable individuals who helped shape both my work and my personal growth. I would like to take this opportunity to express my deepest gratitude to all those who supported me during my Ph.D. studies.

First and foremost, I am enormously grateful for the mentoring and guidance provided by my supervisor, Prof. Olga Fink. I will always cherish the memory of receiving the offer to be part of the IMS team, and the journey that followed was both challenging and wonderful. I deeply appreciate the trust, patience, encouragement, and constructive feedback that Prof. Olga Fink provided, which was instrumental in my growth and success.

I also had the pleasure of collaborating with Dr. Qin Wang, Dr. Manuel Arias Chao, and Dr. Minghao Han, whose precise suggestions were invaluable to my thesis. I am extremely grateful for all the help they offered.

My appreciation extends to all members of the IMS/IMOS team for their discussions, support, and camaraderie during my doctoral journey. In particular, I would like to thank Katharina Rombach, Ismail Nejjar, and Zhichao Han, who offered me valuable perspectives and assistance. It was an honor to spend my Ph.D. years with such a fantastic team, and I learned so much from each of them.

I would also like to express my gratitude to Prof. Zhiwu Huang, Prof. Giovanni Sansavini, Prof. David Coit, and Prof. Ioannis Anastasopoulos for serving on my Ph.D. committee. It was an honor to have such distinguished scholars evaluate my work.

Lastly, I would like to express my gratitude to all who made my stay in Zurich memorable, particularly Jiemin Du, whose unwavering support was invaluable.

And most importantly, I would like to express my deepest gratitude to my family members, particularly my mother Kuiran Bao, my father Yingping Tian, and my grandmother Prof. Jiazhen Fu. Their unconditional love and support gave me the confidence and courage to pursue this academic path, and they have always been my source of strength.

In conclusion, this thesis is the result of the contributions of many people, both direct and indirect. While it is impossible to name everyone, I appreciate all the help I have received along the way.

1 Introduction

1.1 Motivation

Prognostics and Health Management (PHM) focus on providing insight into failure mechanisms, developing optimal maintenance policies, and refining management approaches over the entire life-cycle of industrial systems in order to achieve high performance, reliability, and availability with the lowest cost under their distinct operating and degradation conditions (Pecht, 2009). Recently, there have been several advancements in PHM applications, in particular in fault detection (Michau et al., 2020; Michau and Fink, 2021), fault diagnostics (Wang et al., 2019a, 2021), and prognostics, i.e., remaining useful life (RUL) prediction (Chao et al., 2022). However, PHM does not end with prognostics. Based on the information from diagnostics and prognostics, as well as available resources and operational demand, system health management aims to provide maintenance decisions or tactical operations. If accurate RUL predictions are available, domain experts make decisions to forestall the failure and schedule corresponding fault mitigation actions. Fault mitigation actions include operational failure avoidance, failure recovery (low-level control, re-configuring, re-planning), and preventive maintenance or repair. However, the decisions often depend on the experience and expertise of domain experts and may be difficult to take efficiently and effectively due to the number of possible options and the resulting consequences.

While predicting the remaining useful lifetime and scheduling timely maintenance actions may already improve the availability of the systems, it does not aim to prolong the remaining useful lifetime or to influence it proactively. Prescriptive maintenance, however, goes beyond the predictions of failure time. It aims to prescribe optimal operation with respect to the usage or lifetime of the asset and integrate the degradation into the control. Prescriptive maintenance helps to proactively manage the system's performance, reliability, and availability, thereby reducing maintenance costs and prolonging the system's remaining useful lifetime (Cho et al., 2022).

Prescriptive maintenance is a very promising and urgently required research direction, not only for industrial applications but also for infrastructure systems due to the growing complexity and increasingly demanding requirements on their performance and availability (Garrone et al., 2023; Gordon and Pistikopoulos, 2022; Cho et al., 2022). However, this research direction is relatively recent and has not been explored extensively. Previous applications have mostly focused on mid-term or long-term maintenance policies (Liu et al., 2019a) and maintenance planning (Meissner et al., 2021; Matyas et al., 2017) by using optimization-based (Cho et al., 2022) or heuristic methods (Meissner et al., 2021); see Figure 1.1 below. Most importantly, previous prescriptive maintenance algorithms only performed one-step decision-making, which optimized or derived the maintenance plan and repeated this process in the next maintenance cycle (Meissner et al., 2021; Cho et al., 2022). Such approaches do not consider the subsequent impact of the decisions and the dynamic degradation process. However, if we are aiming to take realistic dynamic degradation processes into account for the maintenance decisions, sequential decision-making is required (Kanso et al., 2023; Björsell and Dadash, 2021).

Previous works have tried to integrate the degradation conditions into decision-making by model-based control, such as health-aware control (Hu et al., 2019; Yin and Choe, 2020) or fault-tolerant control, which incorporate the effects of aging, fatigue, and damage of the considered components into the objective function (Kanso et al., 2023; Björsell and Dadash, 2021; Meyer and Sextro, 2014). However, there are still limitations when applying these

methods to prescriptive maintenance tasks. Model-based control, such as the previously mentioned model predictive control approaches (MPC), typically requires an explicit model of the degradation behavior, which is limited in its application to systems with known or simple degradation behavior. Besides, model-based approaches also suffer from high computational costs due to the online optimization process, which becomes more time-consuming in high-dimensional control tasks. Moreover, all of the previously applied methods rely on explicit health features, which, unfortunately, are often unavailable or difficult to derive. Besides, since the degradation process is, in fact, stochastic in nature, the model-based approaches are typically vulnerable to these uncertainties. These challenges precluded the application of immediate-level maintenance operations for complex systems. Thus, sequential decisions for prescriptive maintenance when the explicit degradation features are not directly available or cannot be inferred easily remain an open research question. Examples of such cases include distributing the load in real-time to prolong the working cycle and remaining useful life of multi-power source systems without any explicit information on the health features.

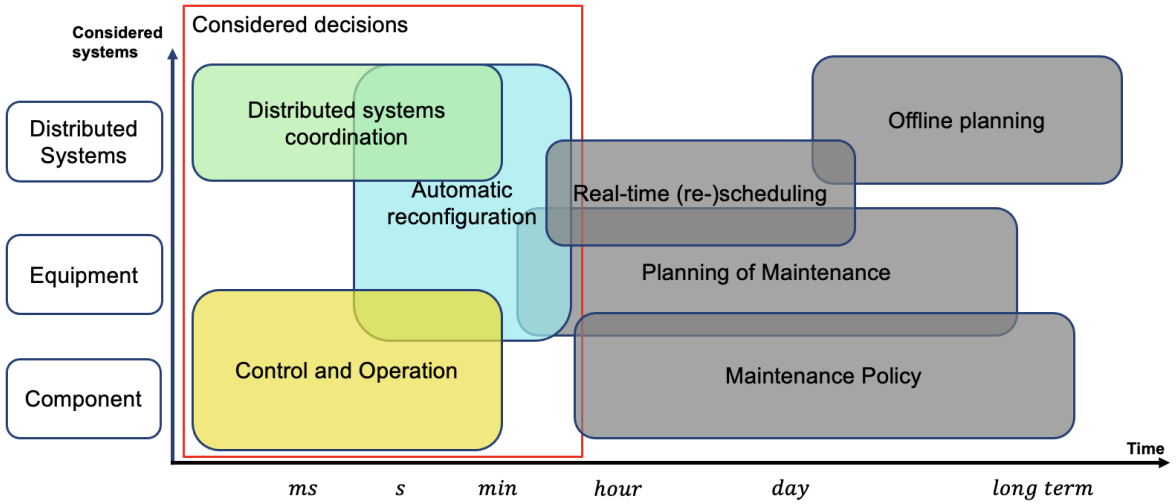

Figure 1.1: Decision types in Prognostics and Health Management, modified from (Nicod et al., 2017).

In addition to model-based control, one potential solution for such sequential decision-making tasks is reinforcement learning (RL). RL algorithms have demonstrated superior performance on various sequential decision-making tasks (Sutton et al., 1992), such as complicated robotics control (Hwangbo et al., 2019), black-box optimization (He et al., 2016; Fawzi et al., 2022), and competitive games (Schrittwieser et al., 2020; Silver et al., 2016). Since the operating, environmental, and degradation conditions are uncertain and are changing frequently, the controller needs to be capable of discovering the underlying dynamics and latent information. RL agents can learn, explore and discover system dynamics by themselves and further optimize given objectives from experience, which is particularly suitable for tasks without known underlying dynamics; see Figure 1.2. In addition, with the development of neural networks, deep RL has shown excellent capability on end-to-end control tasks, integrating data processing and able to perform real-time decision-making based on raw measurements or signals. Moreover, compared to model-based control methods, RL-based methods are particularly suitable for real-time decisions in Figure 1.1 due to their computational efficiency at deployment time (Hwangbo et al., 2019). With these properties, deep RL becomes a promising alternative to tackle previously unsolvable real-time prescriptive maintenance problems.

Motivated by the achievements of deep RL and the promising potential of immediate-level prescriptive maintenance operations, we aim to develop an RL-based framework to tackle the computational difficulty and the challenge of integrating degradation awareness

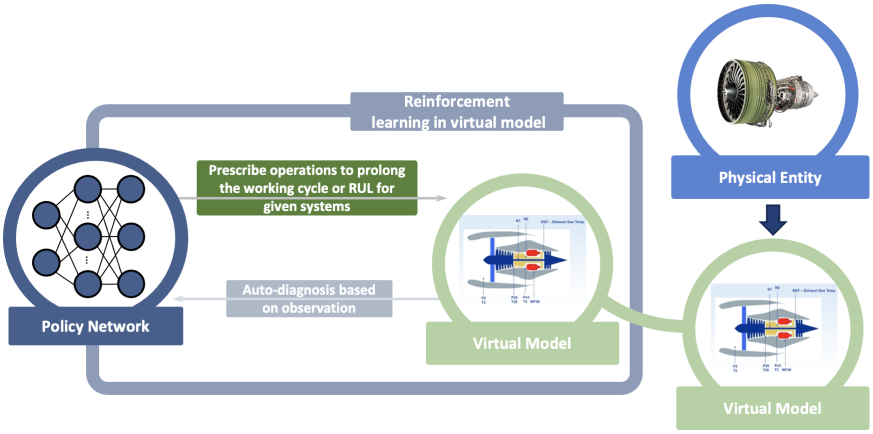

Figure 1.2: Develop reinforcement learning-based maintenance agent in the virtual model.

into sequential decision-making. To achieve this, the main research question we need to answer is the following: *How can we integrate degradation awareness into decision-making and improve system performance and prolong RUL via immediate-level sequential operations?*

1.2 Research gaps and overriding research questions

In this thesis, we propose an RL-based methodology for prescriptive maintenance tasks that can integrate degradation awareness into decision-making and can achieve real-time control to improve the desired performance of industrial systems or prolong the RUL while fulfilling the defined operational requirements. In the following, we elaborate on certain research gaps that need to be addressed.

Real-time model calibration First of all, one of our goals is to integrate degradation into decision support and control. For this purpose, a module capable of deriving the degradation state is needed to provide the necessary understanding of the degradation process. This is important for RL as the agent needs an environment for exploration and interaction to learn the impact of different actions on the degradation state and to learn the optimal policy. Since such explorations cannot be performed in real applications, a simulation environment is required that is able to represent the degradation dynamics properly. Different approaches have been proposed to explicitly model the degradation dynamics, such as cumulative damage models (Hwang and Han, 1986) and the physics-based failure model (Kulkarni et al., 2012). However, explicit models of degradation mechanisms are often only available for systems with rather simple degradation dynamics. An alternative strategy is to capture the macro-level performance impact of the degradation mechanism (Urban, 1973). Under a performance-based modeling strategy, detectable faults, either caused by deterioration or damage, appear as parameter value changes in the model. These types of degradation modeling can provide the mechanism for tracking system behavior under degraded conditions (Kulkarni and Celaya, 2019). Although there have been many successful achievements in system and degradation modeling, it is still a challenge to calibrate the degradation parameters in the performance model in real-time when degradation happens on real systems and the output of the performance model does not match the sensor measurements. Model calibration has been proposed to overcome the gap between the performance model and the measured data. However, previous approaches are computationally expensive and cannot be applied in real time (Borguet, 2012), or else require a large amount of labeled data (Borguet, 2012; Liu et al., 2019d). To overcome these challenges and limitations, we need to answer the research question: *How can we capture the degradation state implicitly in real-time without supervision?*. A well-calibrated performance model can then be used within the simulation

environment for prescriptive maintenance applications. The proposed module can also be applied in real-time, providing a better awareness of the degradation process.

End-to-end load allocation With a calibrated simulation environment that is able to capture the degradation dynamics, we can then develop sequential decision-making strategies based on a calibrated training environment and enable the integration of degradation awareness, as well. We first tackle real-time prescriptive operations that can prolong the working cycle, the usage cycle, or the lifetime of a component or a system. This line of research is comparatively novel, and previous studies have only been considered for relatively simple systems where the degradation can be modeled explicitly. One of the relevant tasks is load allocation on multi-battery systems, which are used in many mission-critical applications such as robotics, and therefore, prolonging their working cycle has recently been gaining more and more importance (Hu et al., 2020). The degradation has a significant impact on the working cycle of multi-power source systems. Thus, prolonging their working cycle can have an essential impact on the deployability of the devices for safety-critical applications. Thus, a good allocation strategy can effectively prolong the discharge and also the RUL, thereby improving availability, maximizing efficiency, and minimizing cost. For this, we consider the problem of prolonging the discharge cycle and also the RUL of batteries in a multi-battery system. In such systems, individual batteries may have diverse degradation states, especially in the second-life battery applications (Peterson et al., 2010; Hu et al., 2020; Fink et al., 2020). In the discharge process, each power source typically starts diverging in its states of health and remaining capacities (Zheng et al., 2015; Severson et al., 2019; Hu et al., 2020), eventually influencing the working cycle of the entire system. Thus, it is important to design proactive allocation strategies that can implicitly infer the degradation conditions and perform actions that respect the individual degradation state in real-time. There has been some research addressing similar tasks. Some previous works (Sui and Song, 2020) prolong the RUL of multi-battery systems by scheduling, which can be considered as a particular case of load allocation. Also, other works focus on the load balancing of batteries, which requires additional algorithms for explicit health features (Ishii, 2021; Chen et al., 2020a), thus limiting the application when such features are unavailable. We therefore address the following research question in this research: *How can we perform prescriptive load allocation based only on raw sensor measurements in real-time?* As discussed above, RL is a promising alternative solution for such end-to-end control tasks. However, it is non-trivial to directly perform previous RL methods on this task. Conventional reinforcement learning methods aim to solve discrete action space tasks or continuous action space tasks without considering constraints. In the load allocation case, continuous action space allocation strategy allows a more fine-grid control, and can thereby improve the performance (Chou et al., 2017). Unfortunately, a general solution in RL for continuous allocation problems with the simplex constraint is still lacking and remains an open research question. Therefore, answering the question *How can reinforcement learning effectively tackle continuous action space allocation tasks?* will help to overcome this limitation.

Distributed systems coordination In addition to the independent decisions that only influence the deployed system, in real-world scenarios, decisions can sometimes implicitly or explicitly influence other systems. For example, in power grids, each decision will inevitably influence the entire network (Rokhforoz et al., 2021). Moreover, in manufacturing systems, each machine is part of a production line, where the maintenance schedule also needs to consider the degradation of other machines and their maintenance operations (Björsell and Dadash, 2021). Such problems can be modeled by multi-agent systems (MAS), and different methods have been proposed. Centralized decision-making methods are one of the solutions,

including methods such as genetic algorithms (Volkanovski et al., 2008; Samuel and Rajan, 2015), particle swarm (Jagtap et al., 2020), heuristic hybrid approaches (Dahal and Chakpitak, 2007), and different optimization-based methods (Xiao et al., 2016; Sadeghian et al., 2019). However, to the best of our knowledge, these methods have mostly focused on mid-term or long-term maintenance scheduling and suffered from high computational costs, which are infeasible for immediate-level operations. On the contrary, decentralized decision-making methods require relatively low computational time and can preserve privacy. Moreover, they also provide more flexibility for modeling. RL-based methods have been proposed as a promising decentralized solution in other application domains. The simplest approach in multi-agent settings is to use independent learning agents (Tan, 1993; Arel et al., 2010). However, when other agents change their policies or when the agents are trained together, the environment becomes non-stationary from the perspective of any individual agent (in a way that is not explainable by changes in the agent's own policy) (Lowe et al., 2017). This gives rise to notorious instability and training difficulties of independent learning (IL) methods (Lowe et al., 2017). Multi-agent reinforcement learning (MARL) is a sub-field of RL, and different methods have been proposed to solve the challenges of IL in MAS (Lowe et al., 2017). However, most of the MARL frameworks focus solely on cooperative or competitive tasks, which is not feasible for some of the maintenance cases, for example, optimizing the maintenance schedule of generating units in a competitive electricity market environment (Rokhforoz and Fink, 2021; Rokhforoz et al., 2021). To address these issues, we aim to develop a general and effective opponent model with a corresponding MARL framework to tackle the non-stationarity and mixed cooperative-competitive objective challenges in MAS tasks. Then the resulting research question that needs to be addressed is the following:*How can we address the non-stationarity of MARL under mixed-objective scenarios?* Any newly developed algorithms need to be compared to other state-of-the-art algorithms to enable a fair comparison of the performance. Unfortunately, there are no openly available (MA)RL simulation environments that are specific to the maintenance tasks. Therefore, to enable a fair comparison of the developed algorithms, we evaluate the performance on the classic differential game and multi-agent particle environments, which include cooperative, competitive, and mixed-objective tasks, and have been a classic and open benchmark to evaluate different methods.

Fast process design and system reconfiguration Besides the discussed sequential decisions, there are also decisions that are unique and are made only once, such as manufacturing process optimization or system (re-)configuration. Many such problems cannot be defined explicitly. For example, the instrumented system design for oil and gas industry processes to improve the reliability (Redutskiy, 2017) and aircraft engine blades designed with contact interfaces to decrease the risk of blade/casing structural contacts (Lainé et al., 2019). In such problems, the analytical forms of some or all of the functions and constraints are unknown or unexploitable. In some cases, the function values can be obtained by simulations, but in some other cases, only examples are available. Most importantly, these functions can be non-differentiable and non-convex, which makes classical gradient-based approaches inapplicable to such problems. There have been some works proposed to address such derivative-free optimization tasks, also as known as black-box optimization (BBO) tasks, which can be categorized into model-based and model-free approaches. Model-based approaches include methods such as Bayesian Optimization (Jones et al., 1998; Rasmussen, 2003), which aims to learn and optimize a surrogate function from samples of the unknown function. However, these type of methods cannot effectively handle parameters with a variable length and are computationally expensive, especially for high-dimensional tasks. Therefore, real-time applications are typically precluded. Recently, reinforcement learning, as a model-free method,

has shown considerable effectiveness on a wide range of BBO tasks (Mirhoseini et al., 2021; Nian et al., 2020), such as chip placement (Mirhoseini et al., 2020) and symbolic regression (Mundhenk et al., 2021) with low computational cost after training, and has become a promising alternative for solving such process optimization tasks. Current reinforcement learning-based approaches mostly formulate the optimization task as a bandit problem or multi-arm bandit problem, which require a large amount of training data and suffer from the large search space and local minima, respectively. These types of challenges are particularly relevant for PHM applications–for example, the optimization of the internal materials ratio of the battery to prolong its working cycle or maximize its capability. In this case, since an explicit system model is often missing, the agent may only have access to a limited number of samples from previous experience or may be able to collect a small amount of data on real systems. In order to enable the RL agent to give a meaningful and reliable recommendation based on limited data, it is necessary to address the following research question:*How can reinforcement learning agents learn and provide recommendations with limited data in large action and state space BBO tasks?* Since there is no specific open benchmark for PHM applications, we evaluate the effectiveness and efficiency on a classical BBO task, the neural architecture search, which is a challenging open benchmark and shares similar data-limited and efficiency challenges with industrial applications.

Overview of research questions

In this thesis, we focus on developing operational strategies integrating degradation awareness to improve the performance of industrial systems or prolong their working cycle and remaining useful life. Besides, since many of the process design and system reconfiguration tasks actually belong to derivative-free optimization problems, we introduce a BBO method, which is especially designed for the data-limited scenarios in real-world. Four modules are proposed in this thesis; two of the modules demonstrate the effectiveness on PHM tasks, while another two modules demonstrate on challenging benchmarks that fulfill the same criteria but can be easily transferred to maintenance problems. To summarize, the following research questions are addressed in this thesis.

1. How can we capture the degradation state implicitly in real time without supervision?
2. How can we perform prescriptive load allocation based only on raw sensor measurements in real-time?
3. How can reinforcement learning effectively tackle continuous action space allocation tasks?
4. How can we address the non-stationarity of MARL under mixed-objective scenarios?
5. How can reinforcement learning agents learn and provide recommendations with limited data in large action and state space BBO tasks?

1.3 Aim and scope

The aim of this research is to *develop an immediate-level prescriptive maintenance framework that integrates degradation awareness into decision-making to improve the system performance, reliability, and availability, and further prolong the working cycle and RUL.*

Contributing to this overall aim, the work is divided into five objectives: (1) Chapter 2 introduces a real-time model calibration module embedded in this framework providing the awareness of degradation processes and calibrating the simulation environments. (2) Chapter 3 introduces an end-to-end load allocation module that can effectively prolong the working cycle and RUL of the deployed multi-battery systems based only on raw measurements;

Chapter 3 also proposes the Dirichlet policy for continuous action space allocation tasks. (4) Chapter 4 proposes a multi-agent coordination module that can tackle the non-stationarity in mixed-objectives scenarios; (5) Chapter 6 introduces an effective and efficient optimization module that can provide reliable recommendations with limited data.

1.4 Background

In this section, we briefly present the general concepts of the topics and methods that are in the focus of this dissertation. More detailed background and review of the related works are provided in the respective chapters.

Reinforcement learning Single-agent reinforcement Learning (Sutton and Barto, 2018) aims to develop methods that can effectively learn the operational strategy by interacting with the environment to collect as a higher cumulative reward as possible. And RL algorithms can be mainly divided into two categories, model-free and model-based. Model-based RL agent aims to understand the environment and learn a world model based on its experience and optimize the policy directly from the learned model via optimal control (or combine optimal control methods with value estimation). On the other hand, a model-free RL agent is trying to learn the consequences of its actions, and update the value function and policy progressively without the knowledge of system dynamics. In this thesis, we focus on model-free RL.

RL problems are formulated as Markov decision processes (MDPs), which is a discrete-time stochastic control process. An MDP can be described as a tuple (S, A, r, P, ρ), where S is the set of states that is able to precisely describe the current situation, A is the set of actions, $r(s, a)$ is the reward function, $P(s'|s, a)$ is the transition probability function, and $\rho(s)$ is the initial state distribution. At each time step, the process is in a state s_t and its associated agent chooses an action a_t from the set of possible actions. Given the action, the process moves into a new state s_{t+1} at the next step, and the agent receives a reward r_t. In a general RL setup, an agent is trained to interact with the environment and get a reward from this interaction. The goal is to find a policy π that maximizes the cumulative reward $J(\pi)$:

$$J(\pi) = \mathbb{E}_{\tau\sim\rho_\pi} \sum_{t=0}^{\infty} r(s_t, a_t) \tag{1.1}$$

While the standard RL merely maximizes the expected cumulative rewards, the maximum entropy RL framework considers a more general objective (Ziebart, 2010), which favors stochastic policies. This objective shows a strong connection to the exploration-exploitation trade-off and aims at preventing the policy from getting stuck in local optima. Formally, it is given by:

$$J(\pi) = \mathbb{E}_{\tau\sim\rho_\pi} \sum_{t=0}^{\infty} [r(s_t, a_t) + \beta\mathcal{H}(\pi(\cdot|s_t))], \tag{1.2}$$

where β is the temperature parameter that controls the stochasticity of the optimal policy.

In model-free RL the system dynamics are unknown. Value-based methods and policy-based methods are the two main approaches to solve this problem. The value-based method aims to estimate the state value $V(s)$ or Q-value(state-action value) $Q(s, a)$ during the interactions with environments in order to evaluate the goodness of states and actions and provide guidance for decision-making:

$$V_\pi(s_t) = E_\pi(\sum_{k=0}^{T} \gamma^k R_{t+k+1}|S_t = s) \tag{1.3}$$

$$Q_\pi(s_t, a_t) = E_\pi(\sum_{k=0}^{T} \gamma^k R_{t+k+1}|S_t = s, A_t = a) \tag{1.4}$$

where γ is the discounted factor. And Q-Learning is one of the most classic off-policy value-based methods, which uses a Bellman equation as a simple value iteration update:

$$Q^{new}(s_t, a_t) = (1 - \alpha)Q^{old}(s_t, a_t) + \alpha(r_t + \gamma maxQ(s_{t+1}, a^*)) \tag{1.5}$$

The vanilla Q-learning can only solve discrete state-action space tasks, which limits the application. With the development of deep neural networks (DNN), approaches combine q-learning with DNN, such as Deep Q-Networks (DQN) (Mnih et al., 2015), Prioritized Experience Replay (Schaul et al., 2016), Double DQN (Van Hasselt et al., 2016), etc. And with two innovation technologies, memory buffer (Mnih et al., 2015) and target network (Mnih et al., 2015), value-based methods successfully mastered video games (Mnih et al., 2015) and board games (Silver et al., 2016). Take DQN's Q Network update as an example:

$$\nabla J(\theta) = \mathbb{E}_{\mathcal{D}}[(r + \gamma \max_{a'} \hat{Q}_{\hat{\theta}}(s', a') - Q_\theta(s, a))\nabla_\theta Q(s, a)] \tag{1.6}$$

where $\mathcal{D} \doteq \{(s, a, s', r)\}$ is the replay buffer for storing the MDP tuples (Mnih et al., 2015). For each update, a batch of data samples from the replay buffer to increase the data efficiency, alleviate the effects of data correlation and decrease the variance of the policy update. Also, the use of target network $\hat{Q}$ stabilizes the training.

However, conventional value-based approaches require each action's value for decision-making and value-iteration, which is infeasible for large action space or continuous action space tasks, such as general control tasks.

On the other hand, policy-based methods (Silver et al., 2014) like classical policy gradient (Sutton et al., 1992) can directly optimize action probability without value estimation:

$$\nabla J(\theta) \approx \frac{1}{N} \sum_{i}^{N} (\sum_{t}^{T} \nabla_\theta log \pi_\theta(a_{i,t}|s_{i,t}) (\sum_{t}^{T} r(s_{i,t}, a_{i,t})) \tag{1.7}$$

Compared to value-based approaches, policy-based methods enable continuous action space decision-making but suffer from sample efficiency. In 1.7, every update requires multiple trajectories. To address this issue, the actor-critic framework (Konda and Tsitsiklis, 1999) combines value-based and policy-based methods. The actor-critic method maintains both a policy network and a value network, where the policy network enables sample continuous actions, and the value network provides an estimated cumulative return for policy update without sampling entire trajectories, carrying out relatively good sample efficiency and lower variance of policy gradient. For example, the objective of the policy network for Soft Actor-Critic (SAC) (Haarnoja et al., 2018a) is given by:

$$J(\pi) = \mathbb{E}_{\mathcal{D}} \left[\beta[\log(\pi_\theta(f_\theta(\epsilon, s)|s))] - Q(s, f_\theta(\epsilon, s))\right] \tag{1.8}$$

where π_θ is parameterized by a neural network f_θ, ϵ is an input vector. Unlike 1.7, the gradient signal is from the value network instead of on-policy rollouts. Besides SAC, many algorithms arise under this framework, such as Deep Deterministic Policy Gradient (DDPG) (Lillicrap et al., 2015), Trust Region Policy Optimization (TRPO) (Schulman et al., 2015), Proximal Policy Optimization (PPO) (Schulman et al., 2017), etc. With these advanced methods, deep RL achieves considerable performance on various challenging tasks, ranging from control tasks for robotics (Hwangbo et al., 2019) or Unmanned Aerial Vehicles(UAVs) (Koch et al., 2019) to optimization tasks for chip design (Mirhoseini et al., 2021) or AutoML (Tian et al., 2020a), etc.

Multi-agent reinforcement learning Multi-agent systems have recently found applications in many different domains, which can be formulated as a Markov game (Littman, 1994), also referred to as N-agents stochastic game (Shapley, 1953). For example, the maintenance

problems in large-scale production systems. where the structural and operational dependencies among machines are required to be considered (Su et al., 2022). A Markov game is defined by a tuple $(s_t, a_t^1, ..., a_t^n, r_t^1, ..., r_t^n, p, \mathcal{T}, \gamma)$. Within the tuple, s_t is the state at time step t, a_t^i and $r_t^i = r_t^i(s_t, a_t^i, \mathbf{a}_t^{-i})$ denote the set of actions selected by the policy of agent i and the corresponding rewards assigned to agent i, where the $\mathbf{a}_t^{-i}$ refers to the set of opponent actions. $\mathcal{T}$ is the state transition function, p is the initial state distribution and γ is the discount factor. At each time step t, actions are taken simultaneously by all agents. Each agent aims to maximize its own expected discounted sum of rewards. Thus, for each individual agent i, the objective for its policy π_i can be expressed as:

$$J(\pi_i) = \max_{\pi_i} \sum_{t=0}^{\infty} \mathbb{E}[\gamma^t r^i(s_t, a_t^i, \mathbf{a}_t^{-i})] \tag{1.9}$$

Many works from single agent RL have been adapted into the context of MARL, for example, the monotonic improvement from TRPO to Heterogeneous-Agent Trust Region Policy Optimisation (HATPRO), the maximum-entropy framework from SAC to Probabilistic Recursive Reasoning (PR2), and etc. However, directly transferring single RL methods to MARL setup suffers from the non-stationarity of the adaptive agents. To address this issue, there are two main categories methods, the value decomposition methods, such as QDPP (Yang et al., 2020), QMIX (Rashid et al., 2018), FOP (Zhang et al., 2021c), QTRAN (Son et al., 2019), and VDN (Sunehag et al., 2017), while another type is opponent modeling, such as PR2 and Regularized Opponent Model with Maximum Entropy Objective (ROMMEO). Previous frameworks mostly focues on competitive tasks or cooperative tasks solely, and it is still challenging to tackle tasks where the adaptive agents hold mixed cooperative-competitive objectives.

Degradation The components of industrial and infrastructure assets degrade over time. The degradation process depends on the operating conditions and the internal complex Physico-chemical property (Parry et al., 1995), such as rubbing-wear or corrosion . Degradation processes are, in fact, stochastic in nature, making each system's deterioration trajectory unique (Arias Chao, 2021). To model the degradation process for industrial systems and further develop diagnostics and prognostics technologies, Figure 1.3 illustrates three typical types of the physics-based computational model (Arias Chao, 2021): 1) Models that ignore micro-level degradation processes but captures macro-level degradation characteristics, such as cumulative damage models, which normally utilize the ordinary differential equations (ODE) or empirical time-dependent correlations; 2) Models of the physics of failure, such as thermal and fracture mechanical, which uses numerical methods to solve partial differential equations (PDE) in two or three space variables. 3) Models that capture the macro-level performance impact of the degradation mechanism, which represents detectable faults by parameter value changes in the model and can provide the mechanism for tracking system behavior under different degraded conditions (Kulkarni and Celaya, 2019; Arias Chao, 2021). However, when degradation happens on real assets, there will be discrepancies between model predictions and observation. Without calibration, the virtual model can not represent the real system anymore, and the prognostics become unreliable.

Model calibration In this thesis, we consider the performance model which parameterized the degradation behavior. And in this case, model calibration aims to adjust model parameters that enable computational model dynamics to match reality and to generate model predictions that fit the observations. The inferred model parameters often represent physical quantities that are not directly observable. In the PHM context, degradation calibration is

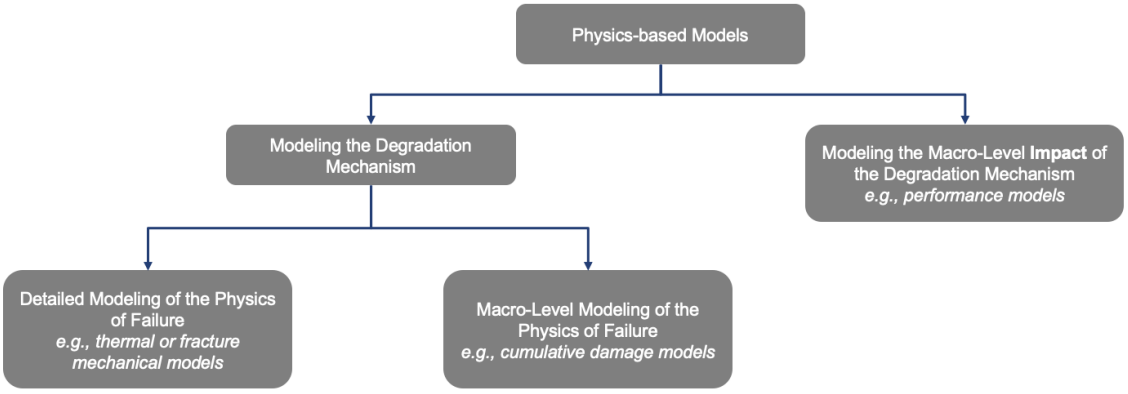

Figure 1.3: Overview of physics-based models

an important application to analyze the discrepancies between model predictions and observations and further infer or identify the degradation conditions. Several methods have been proposed to address the problem of dynamical model calibration. Model-based approaches, such as unscented Kalman filters (UKF) (Julier and Uhlmann, 1997; Turner and Rasmussen, 2010; Borguet, 2012), particle filters (Kantas et al., 2015), Bayesian inference methods using Markov chain Monte Carlo (Rutter et al., 2009; Arias Chao et al., 2015) and Gaussian Process (Kennedy and O'Hagan, 2001; Arias Chao et al., 2015), achieved good results in practical applications. However, approaches of this type all suffer from computational difficulty due to online optimization, which makes them infeasible for real-time calibration of complex models. And data-driven approaches require a large amount of labeled data (Borguet, 2012; Liu et al., 2019d), which limits their implementation in practical applications.

Allocation tasks with reinforcement learning Allocation tasks are to find an optimal distribution of a limited resource given some defined goal and constraints. And the action space of any allocation tasks should be bounded by a simplex constraint. Allocation tasks are very commonly encountered in real-world, such as computational resource allocation (Chen et al., 2020b), task and order allocation (Deng et al., 2020; Feng and Gong, 2020), redundancy allocation (Zhang and Li, 2021; Nath and Muhuri, 2021), portfolio management (Jiang et al., 2017), blockchain applications (Feng et al., 2020), and UAV applications (Shimada et al., 2021). Since these tasks can be formulated by sequential decision-making tasks, RL-based approaches have been proposed to tackle them. However, as far as we know, previous RL methods on such tasks mostly discretize the action space (Tesauro et al., 2006; Ye et al., 2019) or directly apply softmax function on the output (Abrate et al., 2021). The former can not prescribe fine-grid allocation output, and the action numbers increase geometrically with the increase of assets. And the latter suffers from the injective property of probability mapping functions, resulting in less effectiveness and unstableness. Thus, a general solution in RL for allocation problems with the simplex constraint is still lacking and remains an open research question.

Process optimization and system reconfiguration Every engineering system or process is designed with an intended purpose. And in many cases, the engineering design involves tests and experiments since the product or process is not well understood, and the desired performance can not be guaranteed. Such engineering design processes can be seen as an optimization problem. However, the process optimization problems in the industry are mostly derivative-free. There is not a direct or explicit relation between the performance and the de-

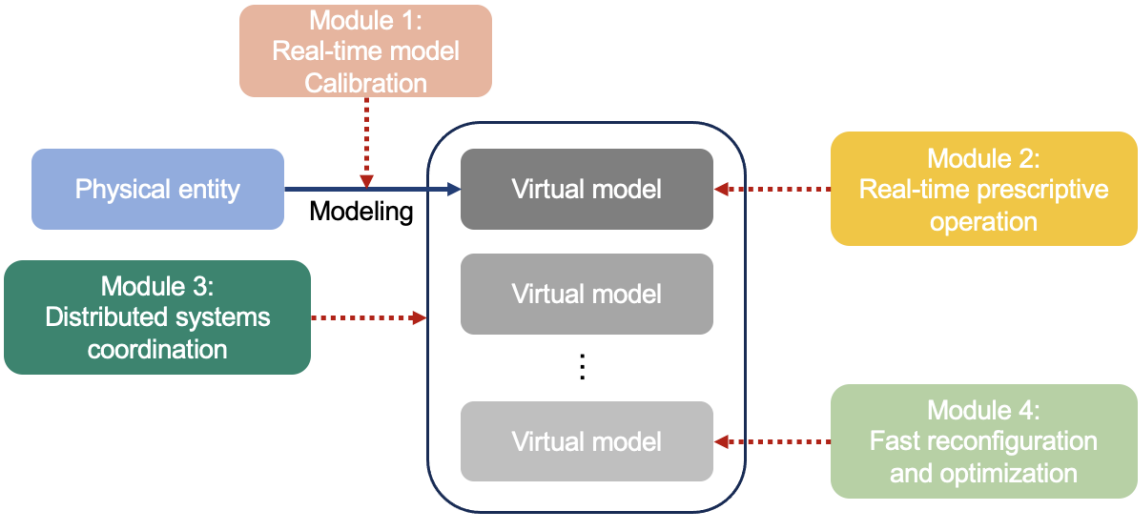

Figure 1.4: Overview of the proposed prescriptive maintenance framework that integrates degradation awareness into decision-making and can deal with three typical immediate-level decision-making scenarios in the industry.

sign, which makes the optimization challenging, and normally relies on the human experience. And After the process optimization, there is still a need of fast system reconfiguration, which is also an optimization problem. For example, in manufacturing factories, the changeover time between productions of two different kinds of products has been calculated by minutes due to the limited profit margins and huge costs caused by the vacancy of production line (Leng et al., 2020). There are several methods have been proposed for such tasks, which can be categorized into model-based and model-free approaches. Model-based approaches, such as Bayesian Optimization (Jones et al., 1998; Rasmussen, 2003), which aims to learn and optimize a surrogate function from samples of the unknown function. However, model-based approaches are computationally expensive, especially for high-dimensional tasks, and can not perform immediately. On the contrary, RL-based methods are able to provide decision support in real-time after training. However, it still requires massive data during training, which limits the applicability in some cases.

1.5 Proposed framework

Following the key motivation to improve performance and prolong the RUL for given systems via real-time maintenance operations, this thesis proposes an RL-based framework that integrates degradation awareness into decision-making and can tackle three typical immediate-level decisions in prescriptive maintenance. We first develop a real-time model calibration module to compensate for the deviation between the simulation environment and the degraded system. We then propose an end-to-end load allocation module for multi-battery systems to prolong their working cycle. And we introduce a novel opponent modeling method with an effective MARL for maintenance coordination problems. Lastly, we provide an efficient optimization module that can be applied to fast process optimization and system reconfiguration tasks. An overview of the proposed framework is shown in Figure 1.4.

Modules

The proposed framework involves the development of four modules with their corresponding methods:

1. *A real-time model calibration module* that aims to calibrate the degradation parameters in the given physics-based performance model in order to compensate for the deviation between sensor measurements and the model output on the degraded system. This

module also plays an important role in guaranteeing well-calibrated training environments for developing prescriptive maintenance applications. Inspired by the connection between control and inference (Levine, 2018), a novel RL-based real-time model calibration module is proposed. The proposed module is evaluated on two open engine datasets. This module can also be applicable to other systems, such as battery systems (Unagar et al., 2021).

2. *An end-to-end load allocation module* that, for the first time, can proactively distribute load demand to each battery based only on raw sensor measurements. This module can effectively prolong the working cycle and remaining useful life of the deployed multi-battery systems. The proposed module is evaluated on four-battery and eight-battery systems with different degradation conditions and can be applied to other multi-power source systems. Moreover, the Dirichlet policy is proposed for the first time to improve the effectiveness and learning stability for continuous action space allocation tasks under simplex constraint, which can be combined with any other RL algorithm to improve the performance when solving allocation problems.

3. *A multi-system coordination module* that can tackle the non-stationary in mixed cooperative and competitive scenarios. In this module, we encode the knowledge that the opponent policies tend to improve over time into opponent modeling. The proposed module provides an effective alternative solution for decentralized maintenance decision-making in industry, which is more flexible, requires low computational cost, and preserves privacy. This module serves as a complement to the previous module when the structural and operational dependencies within each machine or system must be considered.

4. *An effective and efficient optimization module* that can be applied to the process optimization or system reconfiguration problems in industry, especially for the problems in which the analytical forms of some or all of the functions and constraints are unknown or unexploitable. The proposed method demonstrates considerable performance on a challenging black-box optimization task, the generative adversarial networks (GANs) architecture search, and can find a better architecture in far less time compared to the SOTA method.

1.6 Contributions

This cumulative thesis incorporates four published articles in the fields of prescriptive maintenance and machine learning. The key papers are included in Chapters 2 - 5. A summary of their methodology and specific contributions is described in subsections 1.6.1 - 1.6.4.

1.6.1 Real-time model calibration with reinforcement learning

The computational model deviates from the real system due to degradation, which is a very common situation in the industry, and manually adjusting in the laboratory is time-consuming and cost inefficient. Model calibration has been proposed to overcome the gap between the performance model and the measured data. However, previous approaches are computationally expensive and cannot be applied in real-time, or else require a large amount of labeled data. Chapter 2 aims to develop a robust module that can calibrate the degradation parameters of a given physics-based performance model in real-time to compensate for the deviation between the performance model and the degraded real system. This module plays an essential role in the proposed framework. It can not only calibrate the performance model to ensure the training environment matches the degraded system, but most importantly, it provides an indispensable understanding of the degradation state for the subsequent steps of controlling the degradation process. In Chapter 2, we reformulate the model calibration problem as an inverse problem of a tracking problem, which leverages the application of RL for model calibration or parameter inference tasks. As far as we know, it is the first time reinforcement learning that has ever been applied to model calibration or parameters inference tasks. In addition, we propose a novel constrained Lyapunov-based actor-critic (CLAC) algorithm, which achieves superior inference accuracy with considerable robustness. We evaluate the proposed framework on two open engine datasets, the Advanced Geared Turbofan 30,000 (AGTF30) and Commercial Modular Aero-Propulsion System Simulation (C-MAPSS). Dr. Manuel Arias Chao, who contributes equally in this module, fairly and thoroughly compares the proposed RL-based approach with the supervised learning method and model-based unscented Kalman filter method. The proposed method demonstrates considerable applicability and specific advantages on real-time model calibration tasks.

Specific contributions

- A novel Markov decision process formulation is proposed to solve the model calibration or parameter inference problem as an inverse problem of the tracking problem.
- The constrained Lyapunov-based actor-critic algorithm is proposed to improve action stability in calibration tasks.
- A robust real-time RL-based model calibration framework is developed, which can learn to calibrate the physics-based performance model without any labeled data or demonstration during training.

1.6.2 End-to-end load allocation with reinforcement learning

The degradation has a significant impact on the working cycle of multi-power source systems. Thus, prolonging their working cycle can have an essential impact on the deployability of the devices, especially for safety-critical applications. Chapter 3 aims to develop an RL-based prescriptive maintenance module that can proactively allocate the load to different batteries in real time based on raw signal measurements. We evaluate the proposed module on NASA multi-battery models. The proposed module can prolong the working cycle of the deployed systems significantly and shows considerable scalability and transferability. Moreover, we propose the Dirichlet policy to tackle continuous action space allocation tasks, which can be combined with any RL algorithms for a wide range of allocation tasks.

Specific contributions

- A deep reinforcement learning framework for prescriptive maintenance is proposed and evaluated on NASA multi-battery models across different aspects: performance, scalability, and transferability.
- We propose the Dirichlet policy to tackle continuous action space allocation tasks. We demonstrate that the Dirichlet policy is bias-free and provides significantly faster convergence, better performance, and better robustness to hyperparameter changes as compared to the Gaussian-softmax policy.

1.6.3 Multi-agent coordination in mixed cooperative-competitive environments

In addition to the independent decisions, in some cases (e.g., distributed systems), the decisions can implicitly or explicitly influence other systems, which are usually modeled as multi-agent systems. Since the centralized optimization methods are less flexible and suffer from high computational cost, in Chapter 4, we propose a novel decentralized multi-agent reinforcement learning method to address this type of problem. The proposed method follows the decentralized execution mechanism, which is more flexible and computationally more efficient compared to centralized execution for real-time decision-making. Moreover, we propose a general opponent model that allows the agent to tackle mixed cooperative-competitive scenarios, which has been less studied but is widely relevant in real-world applications. This module serves as a complement to the previous module when the structural and operational dependencies within each machine or system must be considered.

Specific contributions

- A novel opponent modeling is introduced to tackle the non-stationarity and mixed-objective tasks in MARL.
- By deriving a lower bound on the log-objective of an individual agent, we further propose an effective MARL method Multi-agent Actor-Critic with Time Dynamical Opponent Model (TDOM-AC). The proposed algorithm demonstrates superior performance on the classic differential game and multi-agent particle environments.

1.6.4 Effective and efficient black-box optimization via reinforcement learning

In Chapter 5, we propose an effective and efficient optimization module, especially for process optimization or system reconfiguration problems in the industry. In such tasks, the analytical forms of some or all of the functions and constraints are unknown or unexploitable, which makes classical gradient-based approaches inapplicable. This work is the final piece of the proposed immediate-level prescriptive maintenance framework, which provides an efficient solution for such optimization tasks. With the novel problem modeling and the introduced adaptive searching strategy, the proposed module can prescribe reliable and reproducible recommendations at the minute level, which is in line with the industrial need. We evaluate the effectiveness and efficiency on a classical BBO task, the neural architecture search, which shares similar data-limited and efficiency challenges with industrial applications and has been an open benchmark to compare to different optimization methods.

Specific contributions

- An RL-based optimization module is proposed for the tasks where the analytical forms of some or all of the functions and constraints are unknown or unexploitable.
- With the novel MDP formulation, the proposed method decreases the action space, improving the optimization efficiency.
- An adaptive learning strategy is proposed to improve the effectiveness of noisy rewards and limited data scenarios.

- We evaluate on GAN architecture search task and find a better GAN architecture with a searching time seven times faster compared to the SOTA method

Chapter 2 addresses the research question: *How can we capture the degradation state implicitly in real-time without supervision?* To answer this question, Chapter 2 proposes a novel deep RL-based module for real-time model calibration and evaluates on two openly available engine datasets. The proposed method can perform real-time degradation parameters inference and achieves superior inference accuracy with considerable robustness to sensor noise and model bias. The proposed method does not require labeled data during training and can be applied to other systems.

Chapter 3 addresses the research questions: *How can we perform prescriptive load allocation based only on raw sensor measurements in real-time?* and *How can reinforcement learning effectively tackle continuous action space allocation tasks?* To answer these questions, Chapter 3 proposes a deep RL-based prescriptive power allocation module to proactively allocate the load to different batteries based only on raw sensor measurements, thereby prolonging the working cycle of deployed multi-battery systems. Moreover, the Dirichlet policy is proposed as an alternative to Gaussian and Gaussian-softmax policy for continuous action space allocation tasks, and we demonstrate its advantages theoretically and experimentally. The proposed module can be applied to other multi-power source systems, and the proposed Dirichlet policy can be applied to a wide range of real-world allocation tasks.

Chapter 4 addresses the research question: *How can we address the non-stationarity of MARL under mixed-objective scenarios?* To answer this question, we first introduce a novel opponent model objective that can successfully alleviate the non-stationarity in opponent modeling and support mixed-objective tasks. And we further derive a lower bound on the log-objective of an individual agent and propose an effective MARL method, Multi-agent Actor-Critic with Time Dynamical Opponent Model (TDOM-AC). In Chapter 5, we demonstrate empirically that the proposed TDOM algorithm achieves superior opponent behavior prediction during execution time. The proposed TDOM-AC outperforms the considered baselines on the performed experiments and considered measures. TDOM-AC results in more stable training, faster convergence, and especially a superior performance in mixed cooperative-competitive environments. The proposed module can be applied to distributed maintenance scheduling or operation tasks where the agents operate in a competitive environment and cannot share their private information.

Chapter 5 addresses the research questions: *Is it feasible and helpful to model a process optimization task as a Markov Decision Process?* and *How can reinforcement learning agents learn and give recommendations with limited data in large action and state space black-box optimization tasks?* To answer these questions, we target a challenging black-box optimization task, GANs architecture search, which suffers from tremendous searching time and data limitations. In Chapter 5, we introduce a novel RL-based black-box optimization method under MDP formulation with an adaptive learning strategy. We experimentally show that the MDP formulation is more effective and efficient compared to the bandit formulation on a given GANs architecture search task, and can target global optimal compared to multi-arm bandit formulation. Additionally, we show that the proposed adaptive learning strategy exhibits noticeable reproducibility under limited data. We achieve state-of-the-art performance on the deployed task and the demonstrated effectiveness, efficiency, and considerable reproducibility are favorable aspects of industrial black-box optimization applications.

Chapter 6 discusses the key findings in the individual works.

Chapter 7 completes this thesis with conclusions and an outlook for future research possibilities.

1.7 Publications

There are four published works that are part of the dissertation:

- Yuan Tian, Manuel Arias Chao, Chetan Kulkarni, Kai Goebel, Olga Fink "Real-time model calibration with deep reinforcement learning." *Mechanical Systems and Signal Processing*, 2022, 165: 108284.
- Yuan Tian, Minghao Han, Chetan Kulkarni, Olga Fink "A prescriptive Dirichlet power allocation policy with deep reinforcement learning." *Reliability Engineering and System Safety*, 2022, 224: 108529.
- Yuan Tian, Klaus-Rudolf Kladny, Qin Wang, Zhiwu Huang, Olga Fink "Multi-agent Actor-Critic with Time Dynamical Opponent Model." *Neurocomputing*, 2023,517:165-172.
- Yuan Tian, Qin Wang, Zhiwu Huang, Wen Li, Dengxin Dai, Minghao Yang, Jun Wang, and Olga Fink. "Off-policy reinforcement learning for efficient and effective GAN architecture search." *European Conference on Computer Vision (ECCV)*, 2020. pp.175-192.

Besides, there are two additional works published during the doctoral study:

- Ajaykumar Unagar, Yuan Tian, Manuel Arias Chao, Olga Fink "Learning to Calibrate Battery Models in Real-Time with Deep Reinforcement." *Energies*, 2021, 14(5): 1361.
- Minghao Han, Yuan Tian, Lixian Zhang, Jun Wang, Wei Pan "Reinforcement learning control of constrained dynamic systems with uniformly ultimate boundedness stability guarantee." *Automatica*, 2021, 129: 109689.

2 Real-time model calibration with reinforcement learning

This chapter corresponds to the published article:[1]

Tian, Yuan, Manuel Arias Chao, Chetan Kulkarni, Kai Goebel, and Olga Fink (2022a). "Real-time model calibration with deep reinforcement learning". In: *Mechanical Systems and Signal Processing* 165, p. 108284.

Abstract: The real-time, and accurate inference of model parameters is of great importance in many scientific and engineering disciplines that use computational models (such as a digital twin) for the analysis and prediction of complex physical processes. However, fast and accurate inference for processes of complex systems cannot easily be achieved in real-time with state-of-the-art methods under noisy real-world conditions with the requirement of a real-time response. The primary reason is that the inference of model parameters with traditional techniques based on optimisation or sampling often suffers from computational and statistical challenges, resulting in a trade-off between accuracy and deployment time. In this paper, we propose a novel framework for inference of model parameters based on reinforcement learning. The proposed methodology is demonstrated and evaluated on two different physics-based models of turbofan engines. The experimental results demonstrate that the proposed methodology outperforms all other tested methods in terms of speed and robustness, with high inference accuracy.

2.1 Introduction

Inference of computational model parameters from real-time measurements can be referred to as *model calibration* (Kennedy and O'Hagan, 2001). Model calibration aims to both obtain model parameters that are theoretically plausible and generate model predictions that fit the observations. The inferred model parameters often represent physical quantities that are not directly observable or observed, i.e., they are not directly obtained from sensor measurements. Therefore, the inference of physics-based model parameters enables one to understand the underlying reasons for a discrepancy between physics-based model predictions and observations, i.e., the *reality gap* (see Figure 2.1). This is of particular relevance for scientific and engineering disciplines where one is interested in improving the physics-based models analytically or explaining the observed processes in light of a given physics-based model structure. Applications can be found in multiple areas, including geology (Elsheikh et al., 2015), climatology (Sansó et al., 2008), biology (Henderson et al., 2009), health (Rutter et al., 2009), finance (Liu et al., 2019d; Deng et al., 2008), cognitive science (Kangasrääsiö et al., 2019), mechanical engineering (Kumar et al., 2013), and applied physics (Higdon et al., 2008).

A particularly important field of application aiming at a reasoned analysis of discrepancies between model predictions and observations is **model-based system health diagnostics** of safety-critical engineered systems. Diagnostics involves detecting when a fault occurs, isolating the root cause, and identifying the extent of the damage (Roychoudhury et al.,

[1] Please note, this is the author's version of the manuscript published in *Mechanical Systems and Signal Processing*. Changes resulting from the publishing process, namely editing, corrections, final formatting for printed or online publication, and other modifications resulting from quality control procedures may have been subsequently added. The final publication is available at https://doi.org/10.1016/j.ymssp.2021.108284.

2013). In model-based health diagnostics, the discrepancy between model and observation is interpreted as a deteriorated or anomalous response of the system. Model-based health diagnostics addresses the diagnostics problem by inferring the value of model parameters, representing the health condition of the sub-components of a system that make the physics-based model predictions fit the observations. In this way, anomalies in the system's behavior are detected and characterized by the value of model parameters.

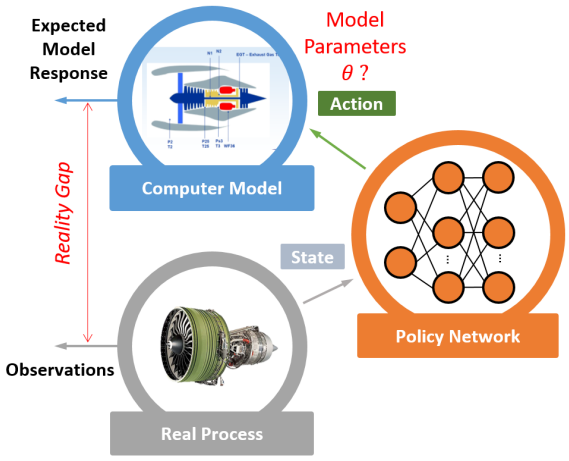

Figure 2.1: **Calibration of physics-based models** aims to infer model parameters that make the physics-based model response follow the observations, thus reducing the *reality gap*. In this work, a reinforcement learning algorithm is used to obtain a neural network policy that bridges the gap between physics-based model predictions and observations in real time.

Because of the relevance of model calibration in applications such as model-based diagnostics, it is important that model calibration provides accurate inference of the model parameters while being robust to uncertainty in the observations and the physics-based model structure. Calibration in real-world scenarios faces **computational and statistical difficulties**. Computational issues are related to the need for running time-consuming simulations using optimisation and inference techniques that generally imply a trade-off between inference accuracy and computation time. Scaling the method to **complex dynamic models** (such as for example flow field calculations), **high dimensional spaces** and **large datasets** further exacerbate the problem. Statistical issues arise from a)the incompleteness of the model representation, b)the existence of multiple solutions, i.e., *confounding solutions* that match the observations, and c)the uncertainty of the observations. Some safety-critical applications, such as model-based diagnostics of aircraft engines considered in this paper, require an inference of the model parameters that is at the same time accurate, robust and is available in real time to enable a fast and reliable state assessment. The necessity of fulfilling all of these requirements at the same time makes the development of methods for reliable dynamical model calibration challenging. Several methods have been proposed to address the problem of dynamical model calibration. When the physics-based model structure is well founded on known physical principles (e.g., aircraft thermodynamic engine models), the majority of the available methods for parameter inference are estimation approaches developed in the fields of **optimal control** (Crassidis and Junkins, 2011) and **statistics** (Sacks et al., 1989). Some examples of popular estimation methods include iterative reweighted least squares schemes (Arias Chao et al., 2015), unscented Kalman filters (UKF) (Julier and Uhlmann, 1997; Turner and Rasmussen, 2010; Borguet, 2012), particle filters (Kantas et al., 2015) or Bayesian inference methods using Markov chain Monte Carlo (Rutter et al., 2009; Arias Chao et al., 2015) and Gaussian Process (Kennedy and O'Hagan, 2001; Arias Chao et al., 2015). Approaches of this type scale relatively well to high-dimensional calibration problems and, with their prob-

abilistic nature, handle observation noise reasonably well. These estimation methods have achieved good results in practical applications and are considered as state-of-the-art methods in several applications, including model-based diagnostics. Yet, despite these attractive properties, they all suffer, at least to some degree, from various model computational and data statistical difficulties in real-world scenarios. In particular, this is because estimation with these methods involves multiple evaluations of the computational model, which makes them unsuitable for real-time calibration of complex models or large datasets. Moreover, these methods are particularly affected by the inadequacy of the physics-based model structure, resulting in an inaccurate characterization of the reality gap.

More recently, **data-driven approaches** have been proposed to calibrate physics-based models. Aiming to avoid time-consuming simulations of previous calibration methods and achieve real-time model calibration, some researchers have proposed alternative approaches to probabilistic formulation of the calibration problem. The most common approach is to address the calibration problem as a supervised learning problem (Liu et al., 2019d). In this case, a neural network algorithm is trained in the inverse relation between the observations and the model parameters. Although these methods provide a real-time calibration approach (only a forward pass over a neural network is required at deployment time), the accuracy of the methods is strongly dependent on the representative quality of the training datasets. As a result, this model calibration approach is not able to adapt to new scenarios without re-training. To mitigate this limitation, an exhaustive mapping of possible system responses under different flight conditions and values of model parameters is required. In practice, in high-dimensional calibration problems with a large range of flight conditions, an exhaustive mapping is infeasible. In addition, such methods can exhibit poor performance in scenarios involving noisy observations, which can limit their implementation in practical applications when no noise mitigation measures are taken into consideration in the learning process.

Because of the issues mentioned above, the real-time, robust, and accurate inference of physics-based model parameters of complex engineered systems remains challenging. However, recent developments in **model-free reinforcement learning (RL)** have fostered a great deal of progress in addressing similar challenges in control problems (Zhang et al., 2020). In fact, RL has proven to be effective in finding optimal control policies for non-linear stochastic systems when the dynamics are either unknown or affected by severe uncertainty (Buşoniu et al., 2018), including complicated robotic locomotion and manipulation (Kumar et al., 2016; Xie et al., 2019; Hwangbo et al., 2019). The policies learnt via RL have the ability to adapt to new scenarios and scale well to large-scale problems at run time. In fact, the decision-making of reinforcement learning can take place through a learned policy without any further optimization or model evaluation, which overcomes the inference speed problem at deployment time. The learned policy can take the form of a neural network or use other function approximation methods.Therefore, model-free RL (Sutton et al., 1992) is a compelling alternative to traditional inference methods for physics-based model calibration.

In this work, we propose a novel formulation of the calibration problem as a **tracking problem** that is modeled by a Markov decision process. Based on this formulation, we apply Lyapunov-based maximum entropy deep reinforcement learning to train an agent that controls the physics-based model parameters to keep the model response matching the observations. In order to overcome the traditional high variance of RL and achieve better robustness to observation uncertainty and model inadequacy, we propose a novel constrained Lyapunov-based actor-critic (CLAC) algorithm. The proposed CLAC algorithm adds constraints on the stability of the policy network and is an extension of the Lyapunov-based actor-critic (LAC) algorithm (Han et al., 2019b).

Without any knowledge of the physics-based model or simulator, the agent explores a large range of possible dynamical responses of the system resulting in good and bad rewards. As a result, the agent is able to exploit the dynamics of the model and produce a robust control (i.e.,

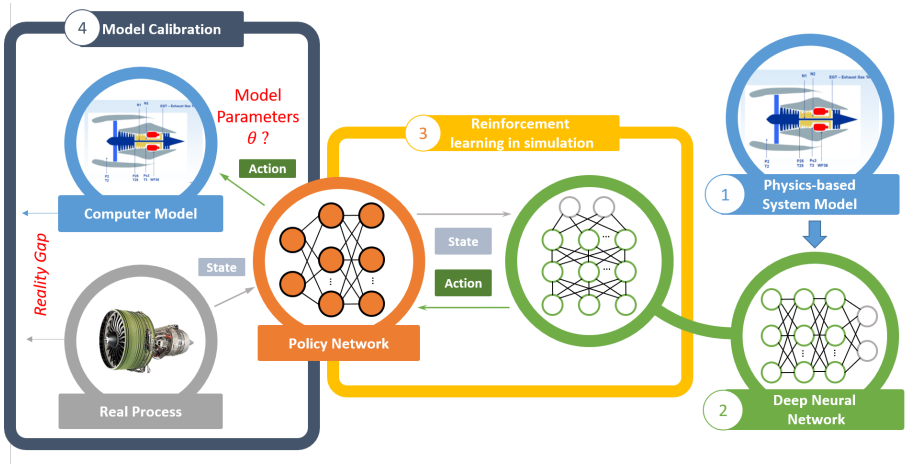

Figure 2.2: **Creating a calibration policy:** Step 1, we identify the parameters of the physics-based model we intend to calibrate. Step 2 (optional), we create a deep neural network (DNN) that models the complex system dynamics. Step 3, we train a control policy using the the physics-based model or the DNN model. **Implementation stage:** Step 4, we deploy the trained policy for real-time model calibration.

calibration) logic. Therefore, the proposed framework overcomes the difficulties of traditional optimal control methods, data-driven approaches and current RL algorithms. It provides: a) an accurate real-time dynamical calibration, b) a policy that can adapt to new scenarios without having been specifically trained on them, c) scalability to more complex systems and high-dimensional spaces, and d) robustness to observation and model uncertainty.

The proposed framework is summarized in Figure 2.2. In the first step, we identify the parameters of the physics-based model that are subjected to inference. In the second step, we use a physics-based model or, alternatively, a surrogate model, in our case a deep neural network (DNN) that emulates the expected system response for measured properties (i.e., observations). In the third step, we use the DNN model to train the calibration policy network via RL. At deployment time, the trained calibration policy is directly deployed to obtain the physics-based model parameters at run time (step 4). The resulting calibration policy is computationally efficient at run time. Most importantly, the calibration policy is robust to uncertainties both in the observations and the physics-based model. The proposed methodology is demonstrated and evaluated on two different physics-based models of a turbofan engine: the Advanced Geared Turbofan 30,000 (AGTF30) and Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) from NASA.

The contribution of this paper is two-fold: 1) We propose a solution to the problem of **real-time dynamic calibration of physics-based models**. In particular, we present a general reinforcement-based model calibration framework that enables real-time inference of system model parameters with a single forward pass through a neural network. While the performance of the framework is demonstrated on two turbofan engine models in this research, it could be easily applied to any system model. Furthermore, the proposed framework does not require any labeled data or demonstration for training. This is in line with the requirements of industrial scenarios where such labels are typically lacking. 2) From the methodological perspective, we propose the **constrained Lyapunov-based actor-critic (CLAC) algorithm**, which provides more action stability, especially on parameter tracking problems, compared to the state-of-the-art LAC reinforcement learning algorithm. This makes the proposed approach robust to noise and high variability, which is again in line with requirements of industrial applications.

2.2 Preliminaries

In this section, we briefly review the basic concepts and notations related to reinforcement learning which is in the focus of the proposed framework.

2.2.1 Reinforcement learning

Reinforcement learning is a sub-field of machine learning that focuses on how an agent interacts with the environment to achieve a specific goal. The environments are typically stated in the form of a Markov decision process (MDP), which provides a mathematical description of decision-making processes. Under the right problem formulation, MDPs can be useful for solving optimization and inference problems, such as the one described above for physics-based model calibration, via reinforcement learning. The details of the MDP formulation of physics-based model calibration will be discussed in Sec 2.3.

In conventional reinforcement learning, an agent is trained to interact with the environment and seek rewards on the basis of its actions. The agent receives a successor state s_{t+1} from the environment as feedback in response to a decision (i.e., action a_t) taken at time-step t. The goal is to find a policy ρ_π that maximizes the discounted cumulative reward $J(\pi)$ (Sutton, Barto, et al., 1998), which is given by the following expression:

$$J(\pi) = \mathbb{E}_{\tau\sim\rho_\pi} \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \tag{2.1}$$

where $\gamma \in [0, 1)$ is the discount factor.

2.2.2 Maximum entropy RL.

The maximum entropy reinforcement learning framework considers a more general objective, aiming to learn a stochastic policy which jointly maximises the expected discounted cumulative reward and its expected entropy $\mathcal{H}(\pi(\cdot|s_t))$ (Ziebart, 2010):

$$J(\pi) = \mathbb{E}_{\tau\sim\rho_\pi} \sum_{t=0}^{\infty} \gamma^t [r(s_t, a_t) + \beta\mathcal{H}(\pi(\cdot|s_t))], \tag{2.2}$$

where β is the temperature parameter that controls the stochasticity of the optimal policy over the reward. Therefore, the resulting stochastic policies balance the exploration-exploitation trade-off and add robustness to the policy.

2.2.3 Stability guaranteed RL.

The maximum entropy reinforcement learning framework can also include a closed-loop stability guarantee of the system dynamics. Such a stability guarantee is particularly relevant when dealing with control problems in real-world applications. Recently, the Lyapunov-based actor-critic (LAC) method (Han et al., 2019b), implementing a stability guarantee, showed state-of-the-art performance on tracking tasks. From a control-theoretic perspective, the task of tracking can be addressed ensuring that the closed-loop system is asymptotically stable. In other words, starting from an initial point, the trajectories of states always converge to a single point or reference trajectory. Therefore, in (Han et al., 2019b), a stability-guaranteed reinforcement learning framework is proposed under the following definition of stability:

Stability Definition. Suppose $c_\pi(\cdot)$ is the cost function, $c_\pi : \mathcal{S} \to \mathbb{R}_+$. The system is said to be mean square stable (MSS) if $\lim_{t\to\infty} \mathbb{E}_{s_t} c_\pi(s_t) = 0$ holds for any initial condition s_0.

Under this definition, the stability objective is given by Equation 2.3. The stability objective defines an energy decreasing condition that drives the trajectory asymptotically to the null space of the cost function, producing predictable behaviour of the agent. Here, we use the Lyapunov function to denote the system's energy, so that the state goes in the

direction of decreasing the value of the Lyapunov function and eventually converges to the origin or a sub-level set of the Lyapunov function.

$$\mathbb{E}_{s\sim\tau}(\mathbb{E}_{s'\sim\rho_\pi}L(s') - L(s)) \leq -\alpha_3\mathbb{E}_{s\sim\tau}c_\pi(s) \tag{2.3}$$

where the α_3 term controls the energy decreasing speed.

2.3 Proposed framework

2.3.1 Model calibration defined as a tracking problem

In this work, we formulate the real-time model calibration problem as a tracking problem, which is modelled by an MDP, and use reinforcement learning to find the optimal tracking policy. We aim to control/infer the model dynamical parameters (θ_{t+1}) to let the model output ($\hat{x_{t+1}}$) match the real sensor measurement(x_{t+1}). However, to solve model calibration by reinforcement learning is not trivial since it is not a conventional control task. To establish a congruence between the model calibration task and the more classical driver tracking task: a driver (equivalent to a computational model) needs to infer the operation (equivalent to physical parameters) of another driver (equivalent to real sensor measurement) in front. To solve this problem, the driver only needs to keep tracking the path of the car in front. Once driver is able to track the front car, the operation he performs, will be the operation he would like to infer. The rationale behind this solution strategy is that learning to track observations of a real system response (x_t) by changing the model parameters (θ_t) results in a control policy that makes the physics-based model yield a sound approximation of the physical process ($\hat{x}_t$), i.e., reducing the reality gap. Consequently, the tracking policy also serves as a calibration policy (Ljung and Gunnarsson, 1990).

Under a tracking solution strategy, the MDP describing the problem is given as the tuple (s, a, r, P, ρ), where state (s) comprises the current model output $\hat{x}_t$, the target value of the system response (observations of the real system) x_{t+1}, and the flight condition w_{t+1}, i.e., $s_t = [\hat{x}_t, x_{t+1}, w_{t+1}]$. The action ($a$) defines the model parameters that need to be calibrated, i.e., $a_t = \theta_t$. The reward/cost function $r(s, a)$ evaluates how good the tracking is. The state transition probability function ($P(s'|s, a)$) corresponds to the dynamics of the system that can be modelled by a physics-based or a surrogate model.

In order to speed up the learning process of the RL algorithm, a discrete time counterpart of the physics-based model F is used. The resulting dynamical system F or a surrogate simulator is modelled by a deep neural network that approximates the dynamic transition equation describing how the expected system response changes given the current observations $\hat{x}_t$, the flight conditions w_{t+1}, and model parameters θ_{t+1}, resulting in:

$$\hat{x}_{t+1} = F(w_{t+1}, \hat{x}_t, \theta_{t+1}) \tag{2.4}$$

For the tracking problem there is, therefore, a desired state that we would like the system to be in at each time step, i.e., x_{t+1}. The task of the agent is to find a control policy $\theta_{t+1} = \pi(\hat{x}_t, x_{t+1}, w_{t+1})$ that minimizes the cost based on a distance metric representing the reality gap of the physics-based model. Here we use the mean squared error (MSE) between the model output $\hat{x_{t+1}}$ and the real system measurement x_{t+1} as the reward signal. In particular, given the dynamical system above and a target system trajectory (i.e, observations), we train the control policy to keep the simulator output matching the real system output by maximizing the cumulative reward as defined in Equation 5.2.

2.3.2 State space and action space

The state s_t describes the current sensor information of the system, the target sensor information of the system and the flight conditions, which is defined as $s_t = [\hat{x}_t, x_{t+1}, w_{t+1}]$. And the action describes as the system parameters need to be calibrated, i.e. degradation parameters, which is defined as $a_t = [\theta_{t+1}]$.

2.3.3 Learning algorithm

In this work, we adopt Lyapunov-based actor-critic (LAC) (Han et al., 2019b) as the learning algorithm, which is based on the soft actor-critic (SAC) (Haarnoja et al., 2018a) algorithm and incorporates a stability guarantee objective. The stability guarantee objective enables a control policy that stabilizes the system in the case of interference by unseen disturbances or uncertainties in the system dynamics. Most importantly, the LAC algorithm has been show to yield the best performance on tracking problems (Han et al., 2019b).

Based on the maximum entropy actor-critic framework, LAC uses the Lyapunov function L_c as the critic in the policy gradient formulation. The objective function of $J(L_c)$ is given as follows:

$$J(L_c) = \mathbb{E}_{(s,a)\sim\mathcal{D}}\left[\frac{1}{2}(L_c(s,a) - L_c^{target}(s,a))^2\right] \tag{2.5}$$

$$L_c^{\text{target}}(s,a) = c + \gamma \arg\max_a L_c(s',a) \tag{2.6}$$

where L_c^{target} is the approximation target for L_c as typically used in RL methods (Mnih et al., 2015; Lillicrap et al., 2015). L_c^{target} has the same structure as L_c, but the parameter is updated through exponentially moving average of weights of L_c controlled by a hyperparameter τ.

The objective function for the policy network is given by:

$$J(\pi) = \mathbb{E}_{\mathcal{D}}[\beta[\log(\pi_\theta(f_\theta(\epsilon,s)|s))] + \lambda(L_c((s', f_\theta(\epsilon,s')) - L_c(s,a) + \alpha_3 c)] \tag{2.7}$$

where π_θ is parameterized by a neural network f_θ, and ϵ is an input vector consisted of Gaussian noise. The $\mathcal{D} \doteq \{(s,a,s',c)\}$ is the replay buffer for storage of the MDP tuples. In the above objective, β and γ are positive Lagrange multipliers which control the relative importance of policy entropy versus the stability guarantee. As in (Haarnoja et al., 2018b), the entropy of policy is expected to remain above the target entropy $\mathcal{H}_t$ (Haarnoja et al., 2018b), here we adopt the same strategy.The value of β is adjusted through the gradient method, thereby maximizing the objective:

$$J(\beta) = \beta\mathbb{E}_{(s,a)\sim\mathcal{D}}[\log(\pi_\theta(a|s)) + \mathcal{H}_t] \tag{2.8}$$

λ is adjusted by the gradient method, thus maximizing the objective:

$$J(\lambda) = \lambda(L_c((s', f_\theta(\epsilon,s')) - L_c(s,a) + \alpha_3 c) \tag{2.9}$$

Under conditions of high sensor noise and simulator bias resulting from an incomplete representation of the system model (i.e., irreducible reality gap), the policy network can exhibit large variance. Such a situation is undesirable in many real-world applications where it is important to obtain a stable or smooth action over time. Therefore, in order to stabilize the action, we introduce the constrained Lyapunov-based actor critic (CLAC) algorithm, a modification of the LAC, which significantly improves the action stability under model uncertainty and sensor noise. In CLAC, the objective function has an additional term that aims to obtain a policy network that has similar optimal action when given a similar or near state (s_{near}) and is given by:

$$\begin{aligned} J(\pi) = &\mathbb{E}_{\mathcal{D}}[\beta[\log(\pi_\theta(f_\theta(\epsilon,s)|s))]+ \\ &\lambda(L_c((s', f_\theta(\epsilon,s')) - L_c(s,a) + \alpha_3 c)+ \\ &\alpha||\pi_\theta^*(s) - \pi_\theta^*(s_{near})||] \end{aligned} \tag{2.10}$$

where α is a positive Lagrange multiplier, and $\pi_\theta^*(s)$ outputs the action with the largest probability. In our case, we use the adjacent time space state s_{t+1} or s_{t-1} to approximate s_{near}.

Input hyperparameters, learning rates α_{ϕ_L}, α_θ
Randomly initialize a Lyapunov network $L(s,a)$ and policy network $\pi(a|s)$ with parameters ϕ_L, θ and the Lagrange multipliers β, λ
Initialize the parameters of target network with $\overline{\phi}_L \leftarrow \phi_L$
for each iteration **do**
  Sample s_0 according to ρ
  for each time step **do**
    Sample a_t from $\pi(s)$ and step forward
    Observe s_{t+1}, r_t and store (s_t, a_t, r_t, s_{t+1}) in $\mathcal{D}$
  end for
  for each update step **do**
    Sample minibatches of transitions from $\mathcal{D}$ and update L, π and Lagrange multipliers with gradients
    Update the target networks with soft replacement:

$$\overline{\phi}_L \leftarrow \tau\phi_L + (1-\tau)\overline{\phi}_L \tag{2.11}$$

  end for
end for

Algorithm 1: Constrained Lyapunov-based Actor-Critic (CLAC)

The entire procedure for training the proposed constrained Lyapunov actor-critic is outlined in Algorithm 1 and the hyper-parameter settings can be found in the Table 2.1

It is worth mentioning that in our experiments the policy is first trained in the simulation environment and then frozen for evaluation.

2.4 Experiments

The proposed framework is demonstrated and evaluated on two different physics-based models focusing on the diagnostics of turbofan engines. The two case studies explore different aspects of real-world calibration problems. Case study #1 corresponds to a one-dimensional calibration problem ($d = 1$) under a wide range of real (i.e., noisy) flight conditions from a small fleet of ten units ($N = 10$). Case study #2 is a more complex system that explores complex failure modes affecting four components of the system simultaneously ($d = 4$). Therefore, case study #2 explores a calibration problem under complex system responses. In contrast to case study #1, case study #2 contains only flight condition from one single unit and, consequently, has a more limited range of flight conditions. More details about the simulator and flight condition data can be found in Appendix.

The performance of the proposed CLAC method is evaluated and compared to two alternative calibration models: a unscented Kalman filter (UKF) and a supervised end-to-end mapping with deep learning algorithm (E2E). The evaluation also covers variants of case study #1 designed to evaluate the robustness of the different methods to uncertainty in the observations and system model predictions.

2.4.1 Neural network architectures and hyper-parameters

The proposed framework requires three neural networks: policy network, Lyapunov network and surrogate network of the dynamic model .

For the policy network, we use a fully-connected multi-layer perceptron (MLP) with two hidden layers of 256 units, outputting the mean and standard deviations of a Gaussian distribution. We adopt the invertible squashing function technique as proposed in (Haarnoja

et al., 2018b) to the output layer of the policy network. For the Lyapunov network, we use a fully-connected MLP with two hidden layers of 256 units, outputting the Lyapunov value. All the hidden layers use leaky-ReLU (Maas et al., 2013) activation function.

The system dynamics (surrogate network of the dynamic model) is approximated with an MLP with four layers ($L = 4$). The hidden layers have 100 units ($n^1 = n^2 = n^3 = 100$). The output layer has the dimension of the sensor reading vector (i.e. $n^L = n$). $ReLU$ activation function was used throughout the hidden layers. For the output layer $\sigma^L = I$ is the identity.

The optimization of the networks' weights was carried out with mini-batch stochastic gradient descent (SGD) and with the *Adam* algorithm (Kingma and Ba, 2015). *Xavier* initializer (Glorot and Bengio, 2010) was used for the weight initializations. Most of the parameters setting are according to the orginal LAC setup (Han et al., 2020), and the target entropy is based on the SAC result (Haarnoja et al., 2018a,b).Table 2.1 provides a detailed overview of the hyperparameters used for the experiments.

Table 2.1: LAC and CLAC Hyperparameters

Hyperparameters	Batch size	LR-Actor/Critic	Target entropy	τ	γ	α_3	Initial β	λ
Value	256	1e-4/3e-4	-d	0.005	0.99	1	2	0.1

2.5 Results

The aim of the proposed framework is to enable accurate, real-time, and robust model calibration for complex systems and large-scale problems. Therefore, in this section, the performance of the proposed method is analysed based on six evaluation criteria: (1)inference accuracy, (2)computational cost, (3)robustness to system model uncertainty, (4)robustness to observation noise, (5)scalability to more complex systems, and (6)tracking accuracy.

Inference Accuracy. The primary objective of model calibration is to infer the values of the model parameters θ. From the application perspective of model-based diagnostics, this objective corresponds to inferring the true underlying degradation parameters. Therefore, we compare the estimated degradation parameters ($\hat{\theta}$) with the ground truth and report the inference accuracy in the form of the root mean square error (RMSE). Table 2.2 shows the inference performance of the unscented Kalman filter (UKF), end-to-end mapping (E2E), and the proposed method (CLAC) in both datasets. With the lowest RMSE, the policy obtained with CLAC shows the best overall performance in both case studies. The improvement is particularly significant under complex fault modes (i.e., Case Study #2). The E2E model yields the worst overall performance in Case Study #1, which highlights the limitations of supervised learning in cases where the flight condition dataset for training is not fully representative of the test conditions. Figure 2.3 shows the inferred unobserved model parameters $\hat{\theta}$ obtained with the three methods in Case Study #1. It is worth mentioning that unlike the end-to-end mapping, which needs the ground truth degradation parameters for training, our framework does not need any prior knowledge about the degradation parameters. This makes the approach more flexible and more applicable to real scenarios.

Since reinforcement learning policies might suffer from a high variance (Henderson et al., 2019), we evaluated the reproducibility of the proposed method by training our agent over five different seeds. As shown in Figure 2.4, we observe that our agent shows a good reproducibility on both tasks.

Computational Cost. One crucial aspect of the proposed method is the ability to perform calibration in real time which is a crucial requirement for real applications. Therefore, we evaluate the time required to perform inference of the model parameters at deployment. Table 2.3 reports the average times required to calibrate a single sample and the total training time

Table 2.2: Overview of the inference performance given by the RMSE between the inferred model parameters and the ground truth with UKF, E2E, and CLAC approaches on complete test trajectories. Best performance is shown in **bold**. [1] The analysis with the E2E model was not performed in Case Study #2 in light of the poor results in Case Study #1

Method	Case Study #1	Case Study #2
UKF	3.42e-04	3.51e-03
E2E	1.36e-03	–
CLAC	**3.30 ± 0.38 e-04**	**2.50e-03**

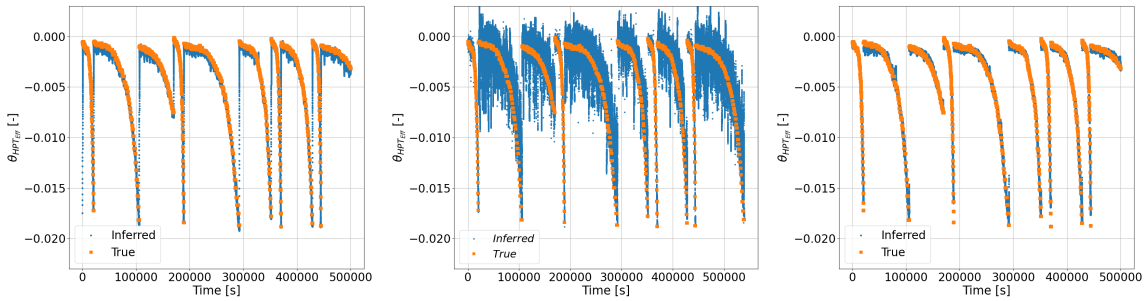

Figure 2.3: Inferred (blue dots) and ground truth (orange squares) traces of θ in Case Study #1 with UKF (left), E2E (middle), and CLAC (right) approaches. The θ values for the ten units are stacked one after the other, generating a single time sequence. Each discontinuity corresponds to the beginning of a new unit. The UKF solution proves a good match to the ground truth with low bias and variance. It is observed that only at the beginning of each unit, the UKF predictions show a large bias. Estimations with the E2E model show a large variance, in particular for units with long degradation profiles. The CLAC method demonstrates a very good match to the ground truth.

with the three methods. In terms of deployment computational cost, the proposed method provides a speed up of $\times 150$ compared to the UKF. Concretely, inference with the proposed CLAC method takes around 40 ms using a CPU thread. This deployment speed is comparable to the E2E model as both methods only require a forward pass over a deep neural network. In contrast, the UKF needs to perform $2 \times (2 \times d + 1)$ model evaluations, which for Case Study #1 amounts to 6 s. The CLAC method requires several hours of training with an ordinary PC and therefore incurs all the computational cost in the training phase, which is typically not critical for practical applications. For real-time applications, the main limiting factor is the deployment time. Therefore, in terms of computational cost, the proposed method has a clear advantage over UKF.

Table 2.3: Overview of the average time required for inference of a single sample with UKF, E2E, and CLAC approaches in seconds [s] for Case Study #1.

Method	UKF	E2E	CLAC
Deployment Time [s]	6	4.2e-02	**4.0e-02**

Robustness to Environment Uncertainty. Robustness to model inaccuracy is an important aspect in model calibration. It is also a well known limitation of model-based methods such as UKF. To evaluate the sensitivity of different approaches to inaccuracies in the models, we apply a model bias to the output of the dynamic system model (i.e., $F(w, \theta) \times \alpha$) to emulate an inadequacy of the system model structure (i.e., inaccurate simulator). We also consider a case where Gaussian noise is added to the dynamic system model, i.e., $F(w, \theta) + \eta$ where $\eta \sim N(0, \alpha_\eta)$. It is worth noticing that adding noise to the output of the simulator

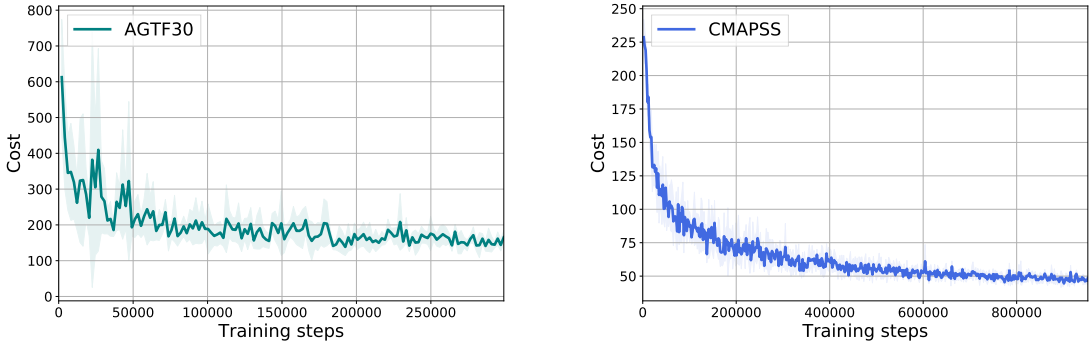

Figure 2.4: The average calibration performance of on AGTF30 and CMAPSS task, where the shaded areas show the 1-SD confidence intervals over 5 random seeds. The X-axis indicates the total training steps, while Y-axis indicates the test return.

transforms the deterministic model into a stochastic system model.

From the RL perspective, the presence of an inaccurate simulator is known as *sim-to-real* transfer. In fact, *sim-to-real* is always a critical problem in reinforcement learning since the agent is trained in a simulated environment which may be different from the real world. In our case, we use a surrogate DNN model to accelerate the training. Therefore, we have an unavoidable error between the DNN surrogate model $\hat{x}_{t+1} = F(w_{t+1}, \hat{x}_t, \theta_{t+1})$ and the engine physics-based model. Then, even in the case where noise is not added, the agent needs to make decisions with noisy DNN model outputs $\hat{x}_t$ at every time step t.

In order to test the trained policy under bias and noisy simulators, we tested two variants where we added a 2% fixed bias (i.e., $\alpha = 1.02$) and a Gaussian noise with a standard deviation of 10% of the model output (i.e., $F(w,\theta)(1+N(0,\alpha_\eta)$ with $\alpha_\eta = 0.1$) to the output of the DNN model. The results in Table 2.4 show that the policy obtained with the CLAC model provides a very good inference even under quite large uncertainty, demonstrating better robustness than the UKF, which failed to optimize a stable inference. The superior inference performance of the CLAC model under 2% fixed bias is visualised in Figure 2.5.

Table 2.4: Overview of the inference performance (RMSE) under model bias (i.e. $F(w,\theta) \times \alpha$) and noise (i.e., $F(w,\theta)(1 + N(0,\alpha_\eta)$) with UKF and CLAC approaches in Case Study #1.

Model Bias: $F(w,\theta) \times \alpha$		
Intensity	UKF	CLAC
$\alpha = 1.02$	2.04e-3	**3.30e-04**
Model Noise: $F(w,\theta)(1 + N(0,\alpha_\eta))$		
Intensity	UKF	CLAC
$\alpha_\eta = 0.1$	#	**4.22e-04**

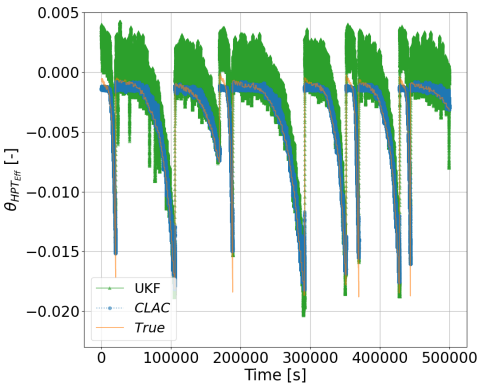

Figure 2.5: Inferred and ground truth (orange squares) traces of θ in Case Study #1 with 2% model bias for UKF (green triangles) and CLAC (blue dots).

Scalability to more complex system and high dimensional model parameters θ. When the dimensionality of the physics-based model parameters θ increases, the complexity of inference increases as well. Due to the non-linear correlation between the degradation

parameters and also between the degradation parameters and observations, the solution of the calibration problem in high dimensional spaces can lead to *confounding* solutions. In scenarios with noisy observations and systems with poor observability, the solution of inverse problems, such as UKF methods, might involve a spurious association of calibration factors that have similar system outputs. To test the scalability of our policies, we performed experiments on controlling 1, 2, and 4 degradation parameters in AGTF30 experiments (i.e. Case Study #2). Figure 2.6 shows the inferred and ground truth traces of a four-dimensional θ in Case Study #2 with UKF (left) and CLAC (right) approaches. As in the previous plots, the θ values for 1315 fault intensities are stacked one after the other, thus generating a single time sequence. We can observe that the UKF solution does *confound* or *smear* the source of degradation. Moreover, as observed in Case Study #1, at the beginning of each fault combination, the predictions show large bias. Both of these issues are efficiently solved with the proposed CLAC method.

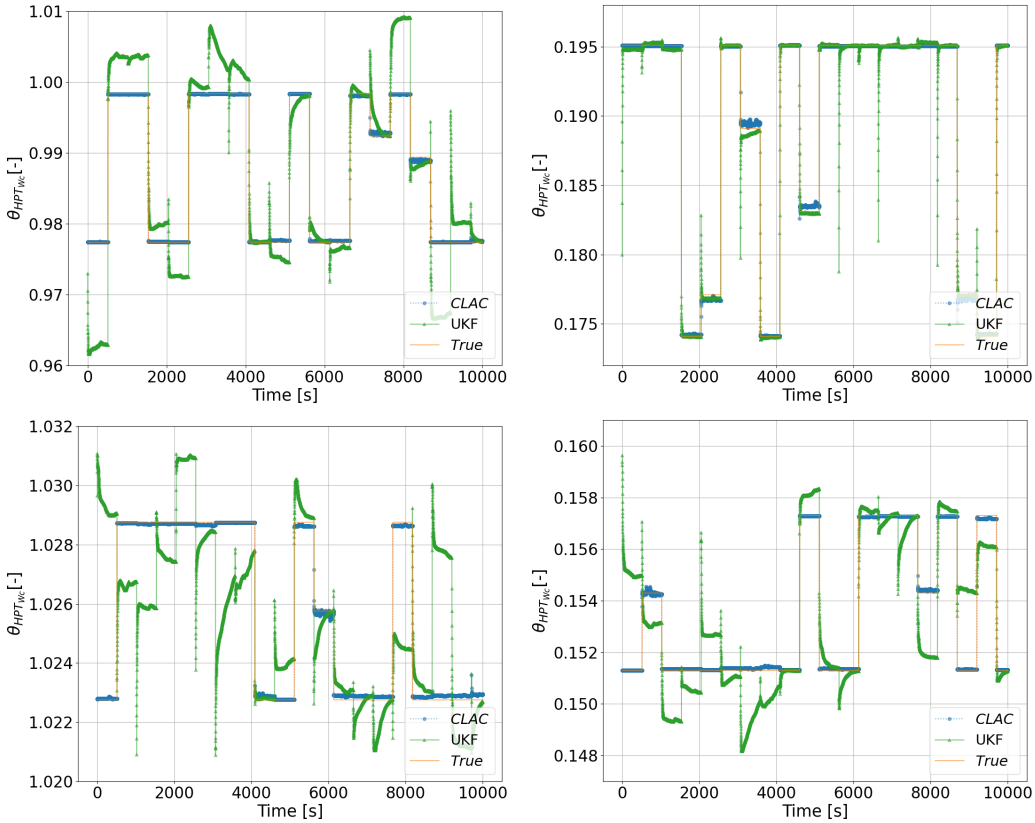

Figure 2.6: Comparison to the ground truth in four θ parameters (HPT and LPT flow (i.e., Wc) and efficiency (i.e., Eff.)) and different degradation parameter settings. The orange solid line is the ground truth degradation parameter value in different trajectories. The blue dotted line is the policy's action with CLAC and the green solid line with triangles is the UKF prediction.

Robustness to sensor noise. In real scenarios, the observations are always noisy. Therefore, it is also important to obtain a policy that is robust to sensor noise. To evaluate this effect, we modelled the engine sensor noise and generated a noisy flight condition by adding Gaussian noise with an intensity of 70 db signal to noise ratio ($SNR_{db} = 70$) to the original flight condition. Table 2.5 shows the impact of noise on the inference performance of the UKF and CLAC methods. In this case, although our policy still shows good inference ability, UKF is more robust to sensor noise.

Tracking accuracy. We proposed to formulate the calibration problem as a tracking problem and use reinforcement learning to track the operational trajectories of the real systems (i.e., the observations) while being constrained to have a stable policy. Therefore, we

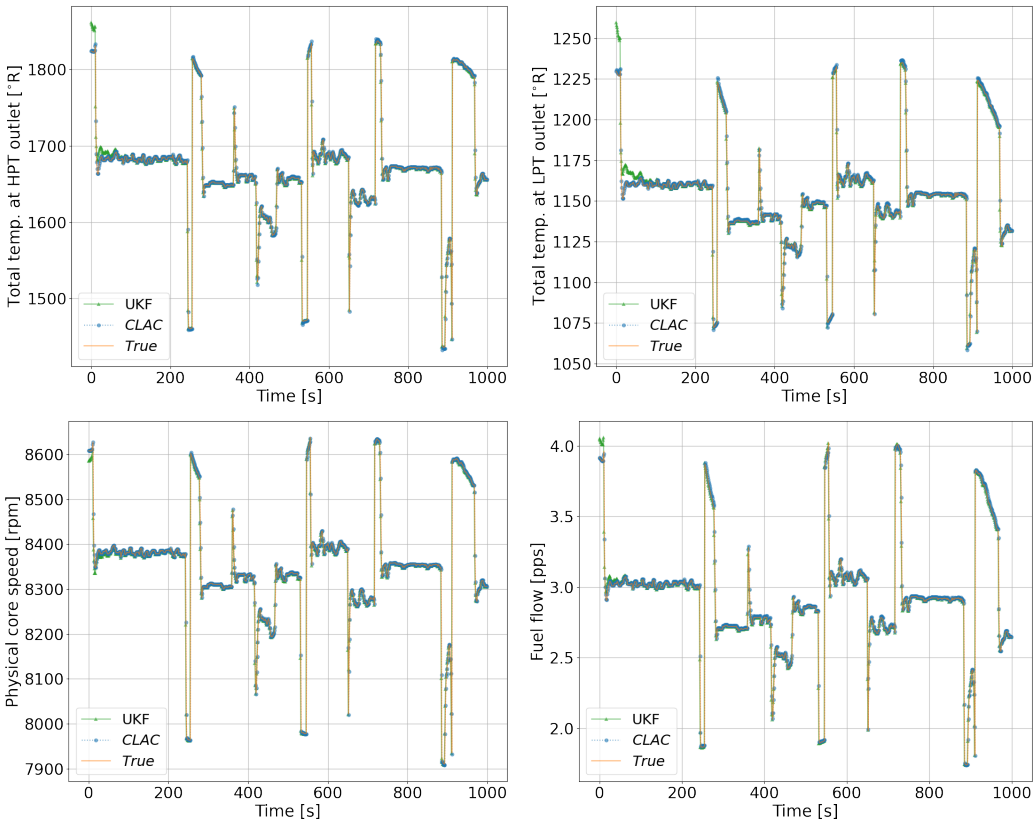

Figure 2.7: Tracking with the proposed CLAC method (blue dots) and UKF (green triangles) on a subset of Case Study #1 for four sensor outputs. Ground truth is shown with the orange solid line. The CLAC policy is able to keep all these sensors on track.

Table 2.5: Overview of the inference performance under observation noise with UKF and CLAC approaches for complete test trajectories - System #1.

Observation Noise: $x_s + \epsilon$		
Intensity	UKF	CLAC
$\text{SNR}_{db} = 70$	**3.72e-04**	7.18e-04

evaluate the error between the observed real system response and the calibrated model output. Figure 2.7 shows that our policies exhibit a good tracking ability for the model outputs. Table 2.6 provides a complete overview of the RMSE for each of the evaluated test cases. Although the CLAC framework demonstrates a good tracking ability in all the setups, the UKF achieves an even better tracking. This difference in the performance between UKF and CLAC is expected since in the RL agent is actually solving a more complicated problem. In particular, the current state contains the output of the DNN model $\hat{x}_t$ instead of the historical observation (x_t). As a result, small errors accumulate affecting the tracking performance. However it is worth point out that precisely this aspect ensures that the proposed policy action generalizes well to unseen degradation trajectories.

2.5.1 Ablation study

Comparison between LAC and CLAC algorithms In this research, we propose to extend LAC to CLAC to improve the stability of the policy under noisy conditions. To demonstrate the benefit of the proposed extension, we compare the inference performance of both algorithms, LAC and CLAC, on Case Study #1. In the C-MAPSS experiments, the

Table 2.6: Overview of the tracking performance given by RMSE with UKF and CLAC approaches in both case studies.

Method	Case Study #1	Case Study #2
UKF	**0.62**	**1.78**
CLAC	0.98	5.54

flight conditions are very diverse and the DNN model is not very accurate and is particularly noisy. Therefore, the DNN model may lead to an unstable policy. Figure 2.8 shows the policy's actions with the LAC and CLAC algorithm (orange squares) and ground truth (blue dots) for all the trajectories. CLAC demonstrates a significant reduction in the variance of the policy. Concretely, in terms of the RMSE metric, the LAC results in a RMSE of 1.3e-03 while the CLAC leads to an RMSE of 3.3e-04. Therefore, CLAC provides a 4× inference improvement.

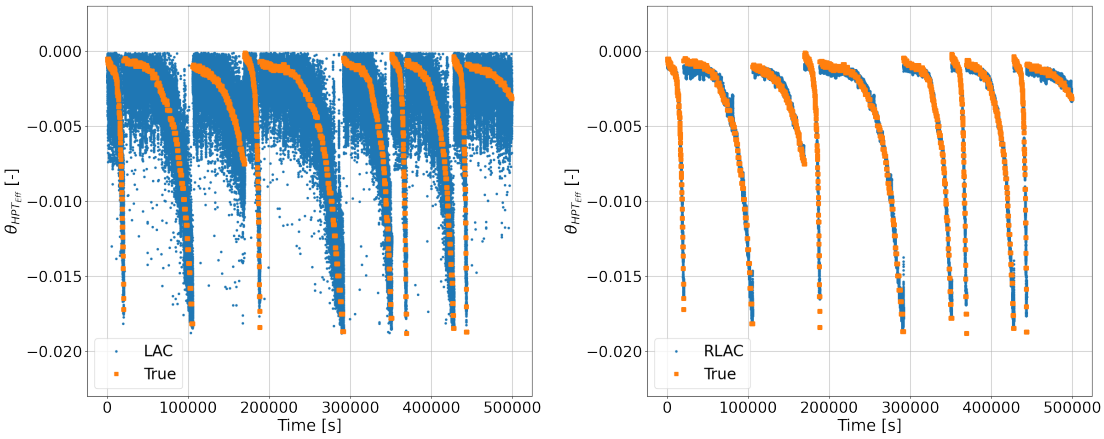

Figure 2.8: **Left:** Inferred policy's actions with LAC algorithm (blue dots) and ground truth (orange squares) in Case Study #1. The trajectories of the ten units, stacked one after the other, are shown. SAC policy exhibits quite large variance. **Right:** Inferred policy's actions with CLAC algorithm (blue dots) and ground truth (orange squares). the CLAC policy shows a good stability and a very good match to the ground truth.

2.6 Conclusions and future work

In this research, we proposed a maximum entropy reinforcement learning framework and the constrained Lyapunov-based actor-critic (CLAC) algorithm for model calibration. The proposed calibration methodology achieves an instantaneous, high inference accuracy and robustness that makes the proposed methodology applicable to noisy, and large-scale calibration problems in real-time. This capability was achieved purely on the basis of training in a simulation environment without any tedious sampling or computationally expensive solution of an inverse problem. Moreover, and in contrast to the end-to-end learning architectures, the proposed methodology only requires access to the model and the observations, eliminating the need for any ground truth calibration parameters for training. Overall, the proposed CLAC algorithm achieves more precise and faster inference than the prior state-of-the-art approaches while being more robust to system model uncertainty.

The proposed framework can be generally combined with various RL algorithms, or can even be extended to the meta RL (Rakelly et al., 2019; Finn et al., 2017) or hierarchical RL (Dietterich, 2000; Barto and Mahadevan, 2003). All our experiments are currently performed in a simulated environment. As a next step, we plan to evaluate the resulting policies on real

industrial plants or robots.

Although the learning framework presented in the work is demonstrated in a model-based diagnostics task, it is applicable to any physics-based model, including those used in so-called "digital twins". Therefore, the results presented in this paper suggest a promising research direction in the field of model calibration. From the application perspective, the targeted model-based diagnostics problem was solved using exclusively a set of three deep neural networks. Therefore, the proposed framework is a paradigm shift in the field of model-based diagnostics. Starting with a model-based problem, we demonstrate that a clever arrangement of deep neural networks can learn both the relevant physics of a complex system and the inference techniques required for diagnostics. It is worth pointing out that the use of deep neural networks is very diverse (e.g., functioning as the surrogate of a physics-based system model or as an inference network in a decision-making problem). The proposed framework demonstrates the great potential of fusing physics-based and deep learning models.

3 End-to-End load allocation with reinforcement learning

This chapter corresponds to the published article:[1]

Tian, Yuan, Minghao Han, Chetan Kulkarni, and Olga Fink (2022b). "A prescriptive Dirichlet power allocation policy with deep reinforcement learning". In: *Reliability Engineering & System Safety* 224, p. 108529.

Abstract: Prescribing optimal operation based on the condition of the system, and thereby potentially prolonging its remaining useful lifetime, has tremendous potential in terms of actively managing the availability, maintenance, and costs of complex systems. Reinforcement learning (RL) algorithms are particularly suitable for this type of problem given their learning capabilities. A special case of a prescriptive operation is the power allocation task, which can be considered as a sequential allocation problem whereby the action space is bounded by a simplex constraint. A general continuous action-space solution of such sequential allocation problems has still remained an open research question for RL algorithms. In continuous action space, the standard Gaussian policy applied in reinforcement learning does not support simplex constraints, while the Gaussian-softmax policy introduces a bias during training. In this work, we propose the Dirichlet policy for continuous allocation tasks and analyze the bias and variance of its policy gradients. We demonstrate that the Dirichlet policy is bias-free and provides significantly faster convergence, better performance, and better robustness to hyperparameter changes as compared to the Gaussian-softmax policy. Moreover, we demonstrate the applicability of the proposed algorithm on a prescriptive operation case in which we propose the Dirichlet power allocation policy and evaluate its performance on a case study of a set of multiple lithium-ion (Li-I) battery systems. The experimental results demonstrate the potential to prescribe optimal operation, improving the efficiency and sustainability of multi-power source systems.

3.1 Introduction

Prescribing an optimal course of action based on the current system state, and thereby potentially prolonging its remaining useful lifetime, has tremendous potential in terms of actively managing the availability, maintenance, and costs of complex systems (Ansari et al., 2019, 2020; Popp et al., 2020). In fact, it is a sequential decision-making task that either requires very good dynamical models or models with very good learning capabilities. Reinforcement learning (RL) algorithms have recently demonstrated superior performance on sequential decision-making tasks (Sutton et al., 1992). In particular, model-free RL, which estimates the optimal policy without relying on a model of the dynamics of the environment, has recently yielded very promising results in many challenging tasks across areas as diverse as gaming (Mnih et al., 2013; Silver et al., 2014), control problems (Han et al., 2019a; Tian et al., 2022a), prescriptive maintenance (Meissner et al., 2021) and auto machine learning (AutoML)(Tian et al., 2020a).

[1] Please note, this is the author's version of the manuscript published in *Reliability Engineering and System Safety*. Changes resulting from the publishing process, namely editing, corrections, final formatting for printed or online publication, and other modifications resulting from quality control procedures may have been subsequently added. The final publication is available at https://doi.org/10.1016/j.ress.2022.108529.

An important application of prescriptive operations for multi-power source systems is power allocation with the goal of prolonging the lifetime or the usage time of the systems, thereby improving availability, maximizing efficiency, or minimizing cost. These types of prescriptive operation tasks can be considered as sequential allocation problems. One of the major characteristics of allocation problems is that the action space is bounded by a simplex constraint. This constraint makes the application of RL algorithms in a continuous action space particularly challenging. Besides power allocation (Zhang et al., 2021b), both sequential and single-step allocation tasks are commonly encountered in several other application scenarios, such as task allocation (Deng et al., 2020), resource allocation (Feng et al., 2020; Zhang et al., 2021e), order allocation (Feng and Gong, 2020), redundancy allocation (Zhang and Li, 2021; Nath and Muhuri, 2021) and portfolio management (Jiang et al., 2017). Particularly for allocation tasks involving complex systems, system state and reliability considerations are crucial.

Several research studies have focused on allocation tasks with reinforcement learning (Xiong et al., 2018; Yang et al., 2018b; Jiang et al., 2017; Maia et al., 2020). However, one of the main limitations of the previously proposed RL approaches for allocation tasks is that they were solely able to operate in a discretized action space. This discretization typically precludes, on the one hand, fine-grained allocation actions since the number of discretized actions may become intractably high (Chou et al., 2017). On the other hand, the action space needs to be carefully adjusted if the number of possible allocation options changes. These two aspects substantially limit the scalability of the existing approaches.

To enable more general allocation decision-making, continuous action space is required (Schulman et al., 2017; Haarnoja et al., 2018a). For continuous action-space sequential allocation problems, the RL algorithms need to satisfy the simplex constraints as outlined above. However, the most commonly applied Gaussian policy in other RL tasks is not able to satisfy the simplex constraints (Lillicrap et al., 2015; Schulman et al., 2015, 2017; Haarnoja et al., 2018a). Gaussian-softmax policy could be an alternative solution (Jiang et al., 2017). However, this function is not injective and has additional drawbacks, such as its inability to model multi-modality (Joo et al., 2020). This leads to less efficient training and less effective performance.

In this paper, we focus on continuous action-space sequential allocation tasks and propose a Dirichlet-policy-based reinforcement learning framework for sequential allocation tasks. This enables us to overcome the aforementioned limitations. The proposed Dirichlet policy offers several advantages as compared to the Gaussian, Gaussian-softmax, and discretized policies. The Dirichlet policy inherently satisfies the simplex constraint. Moreover, it can be combined with all state-of-the-art stochastic policy RL algorithms. This makes it universally applicable for sequential allocation tasks. Ultimately, the proposed Dirichlet policy exhibits good scalability and transferability properties. In this research, we theoretically demonstrate that the Dirichlet policy is bias-free and results in a lower variance in policy updates as compared to the Gaussian-softmax policy. Finally, we experimentally demonstrate that the Dirichlet policy provides significantly faster convergence, better performance, and is more robust to changes in hyperparameters as compared to the Gaussian-softmax policy.

The performance of the proposed prescriptive operation framework in the context of sequential allocation problems is evaluated on a case study of multi-battery system applications with the goal of prolonging their working cycles. The developed framework only requires raw, real-time current and voltage measurements, along with the incoming power demand, as inputs. To the best of our knowledge, this is the first time an algorithm has been capable of directly performing the load allocation strategy in an end-to-end way (without the involvement of any model-based state estimation). We will demonstrate that, compared to the equally distributed load allocation, the average length of the discharge cycle of the deployed four-battery system can be prolonged by an average of 15.2% (and an eight-battery system

by an average of 31.9%) over 5000 random initializations and random load profiles, thereby making the batteries more sustainable. Moreover, we will demonstrate that when implemented on degraded batteries in second-life applications with diverse degradation dynamics, the improvement becomes even more pronounced, reflecting a 151.0% extension of discharge cycles on average and thus enabling the reliable usage of second-life batteries.

The contribution of this paper is twofold: 1) We propose a novel RL-based solution for continuous action-space allocation tasks. In particular, we propose the Dirichlet policy and demonstrate its advantages theoretically and experimentally. 2) Based on the proposed Dirichlet policy, we set forth a prescriptive power allocation framework and evaluate its performance on multi-battery systems to prolong the service cycles of these power source systems. The developed framework shows the potential to improve the efficiency and sustainability of power systems with greater effectiveness.

3.2 Related work

Prescriptive operation is a comparatively novel research direction that goes beyond merely predicting the evolution of the system condition and the remaining useful life. The main goal of prescriptive operation is to develop algorithms that are not only able to predict the required measures but also to prescribe an optimal course of action based on the current system state. Different objectives can be considered for prescriptive operation tasks, such as prolonging the remaining useful lifetime and thereby improving the reliability and availability of the system; completing a defined mission or reaching an operational goal, even in the case of adverse conditions or faults; and minimizing emissions and energy consumption. Several research studies have recently taken up the concept of prescriptive operation (Meissner et al., 2021; Consilvio et al., 2019). For example, one investigation,taking economic and environmental impact into account, has prescribed an approach to maintenance operation that improves the efficiency of aircraft maintenance (Meissner et al., 2021). For batteries, optimal charging schedules have been proposed to prolong the remaining useful life (RUL) (Sui and Song, 2020). Prescriptive operation represents a very promising and urgently required research direction with regard to industrial applications due to the rising complexity and increasingly demanding requirements of complex industrial assets(Vater et al., 2019; Ansari et al., 2019). The prescriptive operation problems are, in fact, sequential decision-making problems, for which RL methods have demonstrated very good learning capabilities (Meissner et al., 2021).

In a reinforcement learning task, the agent observes the environment or system state and prescribes an action in order to maximize the cumulative expected future reward. The action space can be discrete, continuous, or mixed. The Q-Learning (Watkins and Dayan, 1992), as well as deep Q-network (DQN) (Mnih et al., 2015) and related variants such as double-DQN (Hasselt, 2010), are normally designed for discrete action-space tasks. To enable continuous action space, policy-based algorithms such as proximal policy optimization (PPO) (Schulman et al., 2017), trust region policy optimization (TRPO) (Schulman et al., 2015), and soft actor-critic (Haarnoja et al., 2018a) have been proposed. These algorithms represent the stochastic policy via a Gaussian distribution and the agent can sample from the distribution to get the specific action. Besides the stochastic policy, the deep deterministic policy gradient (DDPG) (Lillicrap et al., 2015) uses a deterministic policy to tackle the continuous action-space problem. However, DDPG produces a relatively weak performance in complex problems (Haarnoja et al., 2018a). Moreover, beta policy has been proposed to improve the efficiency when physical constraints are present (Chou et al., 2017).

Allocation tasks are very commonly encountered in real-world prescriptive operation problems. However, the application of reinforcement learning to this type of problem and the elaboration of the theoretical perspective have remained relatively unexplored. The task is to find an optimal distribution of a limited resource given some defined goal and constraints. All allocation tasks need to fulfill the constraint that the action space is bounded by a simplex

constraint. Examples of allocation tasks include computational resource allocation, which is highly useful for emerging applications with intense computational resource demands, such as industrial automation (Chen et al., 2020b), blockchain applications (Feng et al., 2020), and unmanned aerial vehicle (UAV) applications (Shimada et al., 2021). Reliability redundancy allocation can help improve system reliability and minimize the cost, weight, or volume (Wang et al., 2020; Sabri-Laghaie and Karimi-Nasab, 2019). Order allocation is becoming increasingly important to commercial enterprises like passenger transportation service companies (Kamandanipour et al., 2020; Cao and Feng, 2020), food delivery services (Sun et al., 2020), and other logistics providers (Jauhar et al., 2021). Optimal allocation directly influences the efficiency and profit of such companies, who are relying on limited resources. In the financial field, portfolio management is, in fact, also an allocation problem (Jiang et al., 2017). Unfortunately, a general solution in RL for allocation problems with the simplex constraint is still lacking and remains an open research question.

A crucial application field of both allocation problems and prescriptive operation is power allocation (Hu et al., 2020) in multi-power source configurations, which has recently been gaining in importance. A major challenge for power allocation strategies for multi-power source systems has been the design of optimal allocation strategies that take distinct observed states into account and consider different dynamics. For example, in multi-battery systems, the individual batteries commonly start diverging in their states of health and remaining capacities through use (Zheng et al., 2015; Severson et al., 2019; Hu et al., 2020). Small dissimilarities at the beginning of the lifetime may be amplified by different usage profiles. Once any of the individual batteries reaches the end of discharge (EoD), the normal operation of the entire system is impacted. Since individual batteries in the system may have dissimilar states of charge that are not directly measurable, distributing the power equally between all batteries is not optimal. Allocating the power demand in an optimal way to each of the individual batteries has the potential to not only prolong the discharge cycle of the entire multi-battery system but also its lifetime, thus improving the sustainability of the batteries.

Different power allocation strategies have been proposed, including rule-based (Wang et al., 2019c; Wang et al., 2019d; Leonori et al., 2020) and optimization-based approaches (Bai et al., 2019; Zhang et al., 2016). There are several limitations to these approaches. In the rule-based load allocation, each specific state would require the definition of customized rules. Thus, the rule-based approaches require extensive prior knowledge as well as extensive experiments for the different conditions that, for example, take into account the state of charge (SoC) or state of health (SoH), which cannot be measured directly. A major drawback of rule-based approaches is that they are typically designed for a specific system and partly for specific usage and operating conditions. Moreover, prior knowledge and model information are also typically required. Therefore, if there is any change in the system configuration or the operating requirements, the allocation rules need to be adjusted. Even for a simple scale-up from four to eight batteries, the allocation rules need to be carefully adjusted. Moreover, since the feedback of such predefined rules is typically delayed, they are also hard to optimize, resulting in sub-optimal solutions.

Optimization-based methods typically require model information. The allocation task is then solved by optimization or control algorithms, such as model predictive control (MPC) (Ishii, 2021; Chen et al., 2020a) and Robust MPC (Huang et al., 2017).These approaches are vulnerable to uncertainties and changes in the schedule of the power profile. Also, to the best of our knowledge, they all rely on extracted information from physics-based models, such as SoC. Furthermore, they are typically computationally intense, especially for high-dimensional allocation problems. Thus, it is challenging for optimization-based methods to deal with complex real-world systems in real time.

Machine learning approaches have been increasingly applied to different battery management tasks, including predicting the future capacity (Nagulapati et al., 2021; Yang et al.,

2018a), SoC (Liu et al., 2021; Ng et al., 2020; Severson et al., 2019; Jiao et al., 2021), SoH, and remaining useful life (RUL) (Xu et al., 2021a). In the power allocation domain, reinforcement learning-based approaches have also been recently investigated in a similar context (Xiong et al., 2018). Compared to the rule-based and optimization-based approaches mentioned above, the proposed model-free RL-based framework provides an alternative solution while overcoming some of their limitations. Unlike rule-based approaches, the strategies for different systems can be learned with model-free RL without any manual feature engineering or prior knowledge. The learned policy demonstrates superior computational efficiency compared to optimization-based methods. This property is particularly important for real-time applications. Moreover, model-free RL is suitable for finding the optimal policy in tasks where the dynamics are either unknown or affected by a large uncertainty (Buşoniu et al., 2018). Under such conditions, the optimization-based methods may fail to find a feasible strategy. Besides, the deep RL typically shows very good performance on end-to-end control tasks and does not require any manual feature engineering. Previous RL-based methodologies addressed power allocation tasks by discretizing the action and state spaces, defining different weight combinations (Xiong et al., 2018; Xu et al., 2021b; Maia et al., 2020). This significantly reduces their scalability and transferability. Due to the exploding action space problem, it is not feasible to directly increase the number of weight combinations for a more fine-grid decision-making (Bellman, 1956; Lillicrap et al., 2015). Thus, to enable a more general power allocation strategy, continuous action space and corresponding approaches (Schulman et al., 2017; Haarnoja et al., 2018a) are needed.

3.3 Preliminaries

A Markov decision process (MDP) is a discrete-time stochastic control process. At each time step, the process is in a state s_t and its associated agent chooses an action a_t from the set of possible actions. Given the action, the process moves into a new state s_{t+1} at the next step and the agent receives a reward r_t; see Fig. 3.1 below:

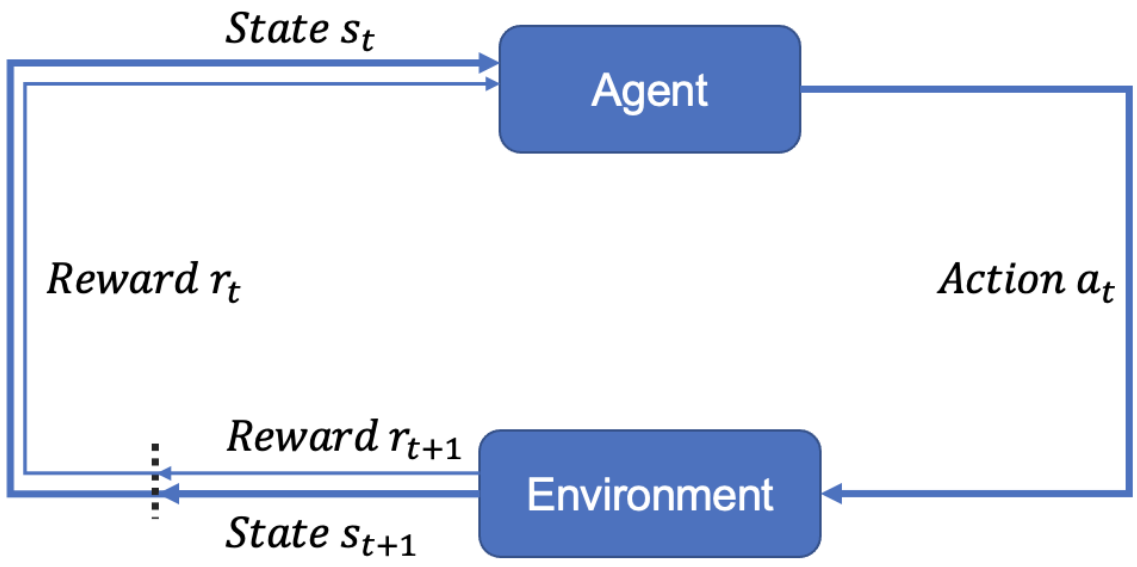

Figure 3.1: Reinforcement learning schematic

An MDP can be described as a tuple (S, A, r, P, ρ), where S is the set of states that is able to precisely describe the current situation, A is the set of actions, $r(s, a)$ is the reward function, $P(s'|s, a)$ is the transition probability function, and $\rho(s)$ is the initial state distribution.

MDPs have been used to describe an environment in reinforcement learning. In a general reinforcement learning setup, an agent is trained to interact with the environment and get a

reward from this interaction. The goal is to find a policy π that maximizes the cumulative reward $J(\pi)$:

$$J(\pi) = \mathbb{E}_{\tau\sim\rho_\pi} \sum_{t=0}^{\infty} r(s_t, a_t) \tag{3.1}$$

While the standard RL merely maximizes the expected cumulative rewards, the maximum entropy RL framework considers a more general objective (Ziebart, 2010), which favors stochastic policies. This objective shows a strong connection to the exploration-exploitation trade-off and aims at preventing the policy from getting stuck in local optima. Formally, it is given by:

$$J(\pi) = \mathbb{E}_{\tau\sim\rho_\pi} \sum_{t=0}^{\infty} [r(s_t, a_t) + \beta\mathcal{H}(\pi(\cdot|s_t))], \tag{3.2}$$

where β is the temperature parameter that controls the stochasticity of the optimal policy.

3.4 Methodology

To solve the continuous action-space allocation tasks, we introduce for the first time the Dirichlet policy. In the following, we first theoretically demonstrate that the Dirichlet policy is bias-free and has a lower variance of policy update as compared to the Gaussian-softmax policy. Moreover, we experimentally demonstrate that the Dirichlet policy provides a significantly faster convergence, better performance, and is more robust to changes in hyperparameters as compared to the Gaussian-softmax policy. Additionally, we combine the Dirichlet distribution with the state-of-art soft actor-critic for the proposed Dirichlet Power Allocation Policy.

3.4.1 Implications of the Gaussian policy

In reinforcement learning, a policy is always required to determine which action to take given the current state. In practice, the stochastic policy is usually parameterized by a conditioned Gaussian distribution $\pi_\theta(\mathbf{x}|s) = \mathcal{N}(\mu_\theta(s), \delta_\theta(s))$, where μ and δ are the outputs of the neural networks. However, the action $\mathbf{x}$ sampled from π_θ is not directly applicable to allocation tasks since the constraint $\sum_{i=0}^{N} a_i = 1$ is not satisfied. It is straightforward to pass the generated candidate action $\mathbf{x}$ to a softmax function $\sigma : \mathbb{R}^N \to \mathbb{R}^N$ to obtain the allocation action:

$$a_i = \sigma(x_i)_i = \frac{e^{x_i}}{\sum_{i=1}^{N} e^{x_i}} \tag{3.3}$$

However, we show in the following that this approach would generate two side effects: a biased estimation and a larger variance. Both of these would jeopardize the policy learning.

Bias

In allocation problems, the policy gradient is written as follows:

$$\mathbb{E}g(\theta) = \mathbb{E}_s \int_0^1 \pi(a|s)\nabla_\theta \log \pi(a|s) Q^\pi(s, a)\mathrm{d}a \tag{3.4}$$

It should be noted that the softmax function is not injective and many possible $\mathbf{x}$ can result in the same action a. More specifically, the softmax function is invariant under translation by the same value in each coordinate, i.e. $\sigma(\mathbf{x} + c\mathbb{1}) = \sigma(\mathbf{x})$ for any constant $c \in \mathbb{R}$. When the softmax function is combined with the Gaussian policy to generate appropriate allocation actions, the distribution of a is, in fact, relevant to the distribution of the candidate action $\mathbf{x}$ and the probability density function (PDF) satisfies

$$\pi(a|s) = \int_{-\infty}^{\infty} \pi_\theta(\mathbf{x} + c\mathbb{1}|s)\mathrm{d}c \tag{3.5}$$

Substituting the above relation into the policy gradient follows that

$$\begin{aligned}&\mathbb{E}g(\theta) = \\ &\mathbb{E}_s \int_{-\infty}^{\infty} \pi_\theta(\mathbf{x} + c\mathbb{1}|s)\nabla_\theta \log \int_{-\infty}^{\infty} \pi_\theta(\mathbf{x} + c\mathbb{1}|s)Q^\pi(s, \sigma(\mathbf{x}))\mathrm{d}\mathbf{x}\mathrm{d}c\end{aligned} \tag{3.6}$$

However, the policy gradient estimator $\mathbb{E}\hat{g}$ used in the ordinary RL algorithm is unaware of the inner integration over the scalar variable c, as in the following

$$\mathbb{E}\hat{g}(\theta) = \mathbb{E}_s \int_{-\infty}^{\infty} \pi_\theta(\mathbf{x}|s)\nabla_\theta \log \pi_\theta(\mathbf{x}|s)Q^\pi(s, \mathbf{x})\mathrm{d}\mathbf{x} \tag{3.7}$$

As the mapping of the candidate action to the allocation action is done in the environment (the specific allocation task), the estimator is created based on the candidate action and inevitably introduces a bias. Even if we assume that the learned critic based on the candidate action can predict the return precisely, i.e. $Q^\pi(s, \mathbf{x}) = Q^\pi(s, \sigma(\mathbf{x})), \forall x$, the bias still exists due to the unawareness of the marginalization over c.

One might also wonder whether using the transformed allocation action a to compute the policy gradient can yield an unbiased estimation. Unfortunately, this is not the case. This would be equivalent to replacing the candidate action x in (3.7) with a. Though it looks similar to the form in (3.4), the distributions π_θ and π are not equivalent. In the end, this will only result in even more biased results.

Variance

In addition to the bias, the Gaussian policy also has the drawback that the variance of the policy gradient estimator is proportional to $1/\sigma^2$. This will induce the variance to reach infinity as the policy converges and becomes deterministic ($\sigma \to 0$) (Chou et al., 2017).

To illustrate this, a useful insight is gained by comparing the policy gradient with the natural policy gradient (Kakade, 2001). The policy gradient in (3.7) does not necessarily produce the steepest policy updates (Amari, 1998), while the natural policy gradient does. The natural policy gradient is given by

$$g_{\text{nat}}(\theta) = \mathbb{E}_s F^{-1}(\theta)\hat{g}(\theta) \tag{3.8}$$

where F denotes the Fisher information matrix, defined as

$$F = \mathbb{E}_{a\sim\pi_\theta}\left[\nabla_\theta \log \pi_\theta(a|s)\nabla_\theta \log \pi_\theta(a|s)^T\right] \tag{3.9}$$

The policy gradient vector is composed of the length and the direction. The ordinary policy gradient may have the correct direction but not necessarily the correct length. The natural policy gradient adjusts the learning rate according to the policy distribution and produces the steepest step. As shown in (Chou et al., 2017), the Fisher information matrix for Gaussian policy is proportional to $1/\sigma^2$, which implies that the more deterministic the policy is, the smaller the update step that should be taken. In the end, the constant update steps will overshoot and increase the variance of the policy gradient estimator.

3.4.2 Dirichlet policy

Since the general Gaussian or Gaussian-softmax policy are not directly applicable to the optimization of allocation problems, applying standard reinforcement learning frameworks or other control algorithms to allocation tasks will result in sub-optimal results that suffer from excessive parameter tuning and/or model complexity. To improve the stability and convergence speed of optimization tasks of allocation problems in continuous action spaces, we propose parameterizing the policy using Dirichlet distribution, which inherently satisfies

the simplex constraint and enables an efficient optimization of allocation tasks in continuous action spaces:

$$\pi_\theta(a|s) = \frac{1}{B(\alpha)} \prod_{i=1}^{N} a_i^{\alpha_i - 1} \tag{3.10}$$

where $B(\alpha)$ denotes the multivariate beta function and can be expressed in terms of the gamma function Γ as follows:

$$B(\alpha) = \frac{\prod_{i=1}^{N} \Gamma(\alpha_i)}{\Gamma(\sum_{i=1}^{N} \alpha_i)}. \tag{3.11}$$

Here, the distribution is shaped by the shape parameters α, which is the output of the neural network $f_\theta(s)$. Thus, the policy is eventually determined by θ.

Bias of the Dirichlet policy

The action a sampled from the Dirichlet policy (3.10) naturally satisfies the constraints on actions in allocation problems, i.e. $\sum_{i=0}^{N} a_i = 1$ and $a_i \geq 0$ (Kotz et al., 2004). Thanks to this property, it is possible to directly sample actions that qualify as allocation actions from the Dirichlet policy, without the need to further constrain them. As a result, the policy gradient estimator $\mathbb{E}\hat{g}(\theta)$ for Dirichlet policies takes the same form as the natural policy gradient $\mathbb{E}g(\theta)$ in (3.4) and is bias-free.

Variance of the Dirichlet Policy

Let $A = \sum_{i=1}^{N} \alpha_i$

$$\log \pi_\theta(a|s) = \log(\Gamma(A)) - \sum_{i=1} \log(\Gamma(\alpha_i)) + \sum_{i=1} (\alpha_i - 1) \log(\Gamma(a_i)) \tag{3.12}$$

Taking the fact that $\partial A / \partial \alpha_i = 1$ and $\partial \alpha_j / \partial \alpha_i = 0$ into account results in:

$$\frac{\partial \log \pi_\theta(a|s)}{\partial \alpha_i} = \psi(A) - \psi(\alpha_i) + \log(a_i) \tag{3.13}$$

with the second order derivative

$$\frac{\partial^2 \log \pi_\theta(a|s)}{\partial \alpha_i \partial \alpha_j} = \psi'(A) - \psi'(\alpha_i)\delta_{ij} \tag{3.14}$$

where $\psi'(z) = \psi^{(1)}(z)$ and $\psi^{(m)}(z) = \frac{d^{m+1}}{dz^{m+1}} \ln \Gamma(z)$ is the polygamma function, the m_{th} derivative of the logarithm of the gamma function.

According to the regularity conditions (Wasserman, 2013), the Fisher information matrix can also be obtained from the second-order partial derivatives of the log-likelihood function,

$$\begin{aligned} F(\alpha) &= -\mathbb{E}_{a\ \pi_\theta} \begin{bmatrix} \frac{\partial^2 \log \pi_\theta(a|s)}{\partial \alpha_1 \partial \alpha_1} & \cdots & \frac{\partial^2 \log \pi_\theta(a|s)}{\partial \alpha_1 \partial \alpha_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \log \pi_\theta(a|s)}{\partial \alpha_K \partial \alpha_1} & \cdots & \frac{\partial^2 \log \pi_\theta(a|s)}{\partial \alpha_N \partial \alpha_N} \end{bmatrix} \\ &= \begin{bmatrix} \psi'(\alpha_1) - \psi'(A) & \cdots & -\psi'(A) \\ \vdots & \ddots & \vdots \\ -\psi'(A) & \cdots & \psi'(\alpha_N) - \psi'(A) \end{bmatrix} \end{aligned} \tag{3.15}$$

The variance of the Dirichlet policy is given by $Var[a_i] = \frac{\alpha_i(A-\alpha_i)}{A(A+1)}$. As the policy becomes deterministic given different states, certain allocation actions α_i and A approach infinity

simultaneously. As shown in (Chou et al., 2017), $\psi'(z)$ goes to zero as z goes to infinity. Thus, the inverse of the Fisher information matrix goes to infinity. This ensures that the update steps will not overshoot and the variance of the policy gradient goes to zero.

To summarize, the Dirichlet policy can intrinsically produce unbiased policy gradient estimations, while the variance of policy updates is also guaranteed to be lower than that of the Gaussian policy. These are both favorable properties to enhance the convergence speed and allocation performance.

3.4.3 Simplex regression experiment

To demonstrate the efficiency and effectiveness of the proposed methodology, we first evaluate it on a simple simplex regression task. The objective is to reconstruct and sequence a 4-dimensional simplex from a 3-dimensional vector obtained by randomly removing a dimension from the target 4-dimensional simplex. For example, given a random simplex vector $[0.4, 0.2, 0.3, 0.1]$, after a dimension is randomly removed, the input data becomes $[0.4, 0.3, 0.1]$. The target output is then the ranked reconstructed simplex $[0.1, 0.2, 0.3, 0.4]$. We use the Mean Average Error (MAE). We apply the proposed Dirichlet policy framework and compare it to the Gaussian-softmax policy. The result shows that the Dirichlet distribution performs better and is more robust to hyperparameters such as, for example, the different learning rates. The Dirichlet policy performs two times better compared to Gaussian-softmax policy with a learning rate of 0.01. In addition, the Dirichlet policy is more robust against different learning rates, while the Gaussian-softmax policy failed with a high learning rate of 0.1. See Fig. 3.2.

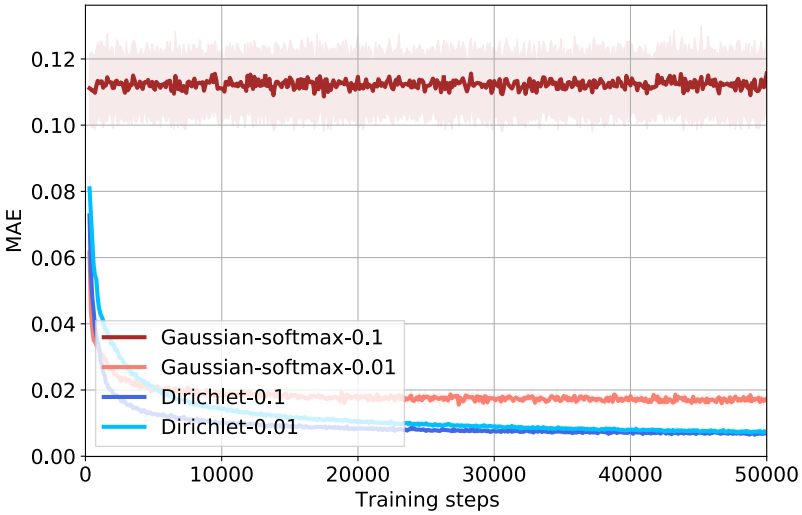

Figure 3.2: **Numerical experiment results.** We compare the learning curves of both output layers with two different learning rates: 0.1 and 0.01, where the shaded areas show the 1-SD confidence intervals over multiple random seeds.

For the neural networks in the numerical experiment, we use a fully connected multi-layer perceptron (MLP) with three hidden layers of 64 units each, outputting the α of a Dirichlet distribution or the μ and σ of a Gaussian distribution. For the Dirichlet distribution network, in the first layer, the Leaky-ReLU activation function (Maas et al., 2013) is applied. In the second layer, the tanh activation function is applied. The α is modeled by a softplus element-wise operation with $\log(1 + \exp(x))$. A constant 1 is added to the output to make sure that $\alpha \geq 1$. The choice of the activation function is motivated by the design of the Beta policy (Chou et al., 2017). For the Gaussian-softmax network, the hidden layers have Leaky-ReLU

(Maas et al., 2013) as the activation function, while the final output layer is mapped to a simplex with a softmax function.

3.4.4 Soft Actor-Critic

In this paper, we applied the off-policy reinforcement learning algorithm soft actor-critic (SAC) (Haarnoja et al., 2018a) with the proposed Dirichlet policy. The SAC is based on the maximum entropy reinforcement learning framework (Haarnoja et al., 2017), where the objective is to maximize both the entropy of the policy and the cumulative return. As a result, it significantly increases training stability and improves exploration during training. Furthermore, it was demonstrated to be 10 to 100 (Haarnoja et al., 2018a) times more data-efficient as compared to any other on-policy algorithms applied to traditional RL tasks.

For the learning of the critic, the objective function is defined as:

$$J(Q) = \mathbb{E}_{(s,a)\sim\mathcal{D}}\left[\frac{1}{2}(Q(s,a) - Q_{target}(s,a))^2\right] \tag{3.16}$$

where Q_{target} is the approximated target of Q:

$$Q_{target}(s,a) = R(s,a) + \gamma[Q_{target}(s', f(\epsilon, s')) - \beta \log \pi(a'|s')] \tag{3.17}$$

The objective function of the the policy network is given by:

$$J(\pi) = \mathbb{E}_{\mathcal{D}}\left[\beta[\log(\pi_\theta(f_\theta(\epsilon, s)|s))] - Q(s, f_\theta(\epsilon, s))\right] \tag{3.18}$$

where π_θ is parameterized by a neural network f_θ, ϵ is an input vector, the $\mathcal{D} \doteq \{(s, a, s', r)\}$ is the replay buffer for storing the MDP tuples (Mnih et al., 2015), and β is a positive Lagrange multiplier that controls the relative importance of the policy entropy versus the cumulative return.

3.4.5 Hyperparameter setting

For the following experiments, we combine the proposed Dirichlet policy with the SAC framework. For the policy network, we use the same architecture design as for the toy experiment with the difference that: 1) 256 units are used and 2) for the additional Q-network, we use a fully connected MLP with three hidden layers of 256 units, outputting the Q-value. All the hidden layers use Leaky-ReLU as the activation function. Fig. 3.3 illustrates the networks. It is worth pointing out that we adopt the same neural architecture as that used in the SAC (Haarnoja et al., 2018a). This neural architecture, typically utilized in other control tasks, has yielded good performance in applications including cart-pole balancing (Haarnoja et al., 2018a), humanoid walking (Haarnoja et al., 2018a), real-world robot control (Mahmood et al., 2018), real quadrotor control (Hwangbo et al., 2017), and others (Schulman et al., 2015, 2017). More information regarding the task-specific input data and output action may be found in Sec. 3.5.

Our implementation exploits the double Q-learning technique (Van Hasselt et al., 2016), whereby two Q-functions $\{Q_1, Q_2\}$ are parameterized by neural networks with parameters ν_1 and ν_2. The Q-function with the lower value is exploited in the policy learning step (Fujimoto et al., 2018), which is useful in mitigating performance degradation caused by the bias in the value estimation.

The optimization of the networks' weights is carried out with the *Adam* algorithm. The *Kaiming* initializer is used for the weight initializations (He et al., 2015). Table. 3.1 provides a detailed overview of the hyperparameters used for the experiments. Training is conducted on a 2.3 GHz 8-core Intel Core i9 CPU.

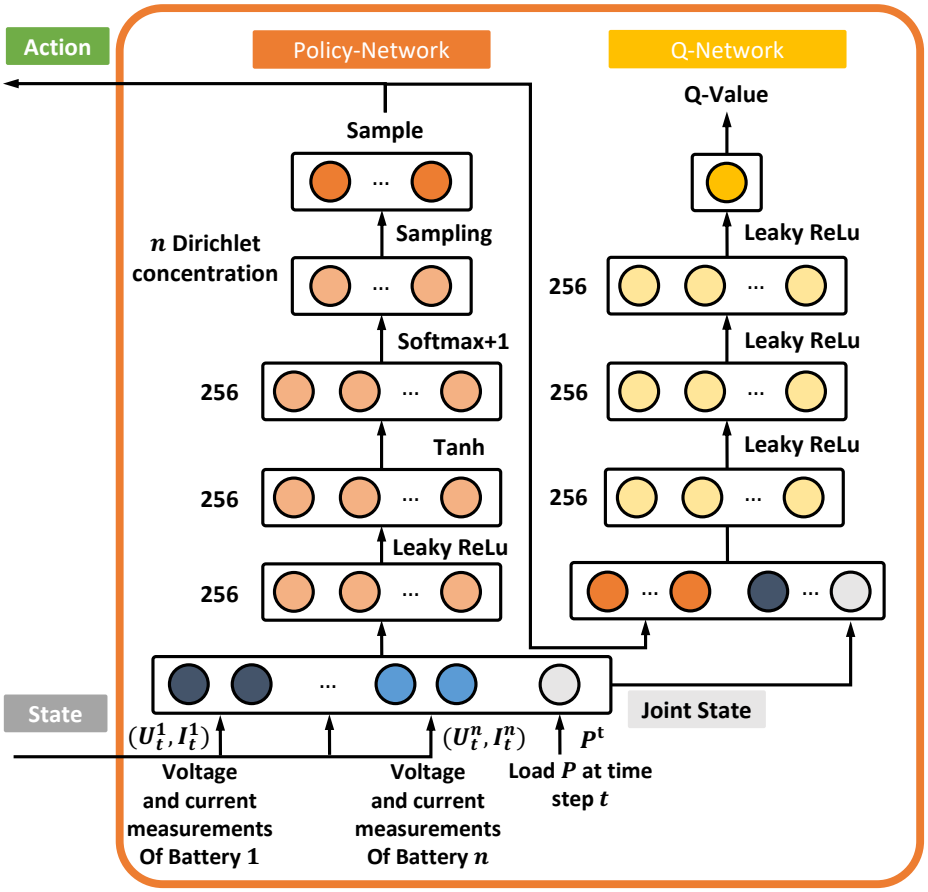

Figure 3.3: Overview of the neural network architectures.

3.5 Power allocation case study

To further evaluate the performance of the proposed method, we design a case study of multi-battery system applications with the goal of prolonging their working cycles. We assume that the power allocation can be controlled at the level of a single battery and that no cell balancing is applied. We would like to emphasize that in this paper, we focus on the algorithm design for the purpose of demonstrating its potential. Implementing this algorithm in real applications would require a dedicated circuit design, which we leave for future work.

The information most commonly used in battery health-related analytics is the operating current and voltage measurements collected by standard battery management systems (Richardson et al., 2018; Severson et al., 2019). In this case study, we aim to utilize only raw measurements of current and voltage directly measured on the batteries (before the DC-DC converter) and extend the capability of machine learning from descriptive and predictive analytics to end-to-end prescriptive decision-making. To the best of our knowledge, this is the first time an algorithm has been capable of directly performing the load allocation strategy in an end-to-end manner (without any involvement of model-based state estimation).

The objective of the desired power allocation strategy is to prolong the working cycle of the deployed multi-battery system. To achieve this, we formulate this problem as a Markov decision process (MDP) and propose to solve it with Dirichlet policy reinforcement learning.

Every operation or maneuver of a multi-battery device will impose a power demand P_t on the system. The RL-based strategy will prescribe an action a_t that dynamically allocates the power demand P_t based on the observed state s_t. In our case, s_t is represented by the real-time operational current and voltage of all the batteries in the system and the total power demand, resulting in $s_t = [V_t, I_t, P_t]$. Then, the system's state changes according to

Hyperparameters	Value
Minibatch size	1024
Learning rate - Actor	1e-4
Learning rate - Critic	3e-4
Target entropy	$-\sqrt{d}$
Target smoothing coefficient(τ)	0.005
Discount(γ)	0.99
Updates per step	1

Table 3.1: SAC Hyperparameters

the allocation strategy and the system dynamics $\mathcal{P}$. To achieve the objective of prolonging the working cycle of the battery device, we provide a reward of $r_t = 1$ to the agent at each time step at which all the batteries in the system are still operational or the voltages are all higher than the end-of-discharge (EoD) state. Given a discount factor $\gamma \leq 1$, an optimal allocation strategy maximizes the expected discounted sum of future rewards, or return:

$$R_\tau = \mathbb{E}[\sum_{t=0}^{\inf} \gamma^t r(s_t, a_t) | s_0 = s] \tag{3.19}$$

where $\mathbb{E}$ indicates the expected value. R_s characterizes the long-term value of the allocation strategy from an initial state s_0 onwards.

Fig. 3.4 provides an overview of the Dirichlet power allocation framework. **A:** When deploying the proposed strategy on any device with a multi-battery system – such as a quadrotor, a robot, or an electric car – any maneuver induces a load demand. For every maneuver, the trained strategy receives the incoming load demand with the real-time current-voltage measurement. It distributes the power based only on the received information or observation, without any online optimization. **B:** The proposed strategy is represented by a neural network, which takes the measurements as input and outputs a weight combination on how the load should be distributed to the individual batteries. The trained network can dynamically allocate the power in an end-to-end way without any estimation of the degradation state. With the input information of current and voltage measurements, it can first implicitly learn the health of the batteries – such as SoC, SoH, or RUL – for decision-making. With the proposed Dirichlet policy, which inherently satisfies the simplex constraint of the allocation tasks, it can prescribe fine-grid allocation weights in a continuous manner and can be trained more efficiently and effectively. **C:** In this paper, the objective is prolonging the working cycle of the deployed multi-battery systems, a goal which could be changed in other tasks according to different requirements.

3.5.1 Simulation environment

We train the allocation strategy in a simulation environment. The simulation environment is a multiple Li-I battery system computational model from the NASA prognostic model library (Daigle and Kulkarni, 2013). It captures the relevant electrochemical processes of the discharge. For an individual battery, the state changes over time as a function of input load and current system states are given by:

$$\begin{aligned} x(k+1) &= f(k, x(k), \theta(k), u(k)), \\ y(k+1) &= g(\mathbf{x}_{t+1}, \theta(k), , u(k), n(k)), \end{aligned} \tag{3.20}$$

where k is a discrete time variable, $x(k) \in \mathbb{R}^{n_x}$ is a state vector, $\theta(k) \in \mathbb{R}^{n_\theta}$ is an unknown parameter vector, $u(k) \in \mathbb{R}^{n_u}$ is the input vector, f is the state equation, $y(k) \in \mathbb{R}^{n_y}$ is the

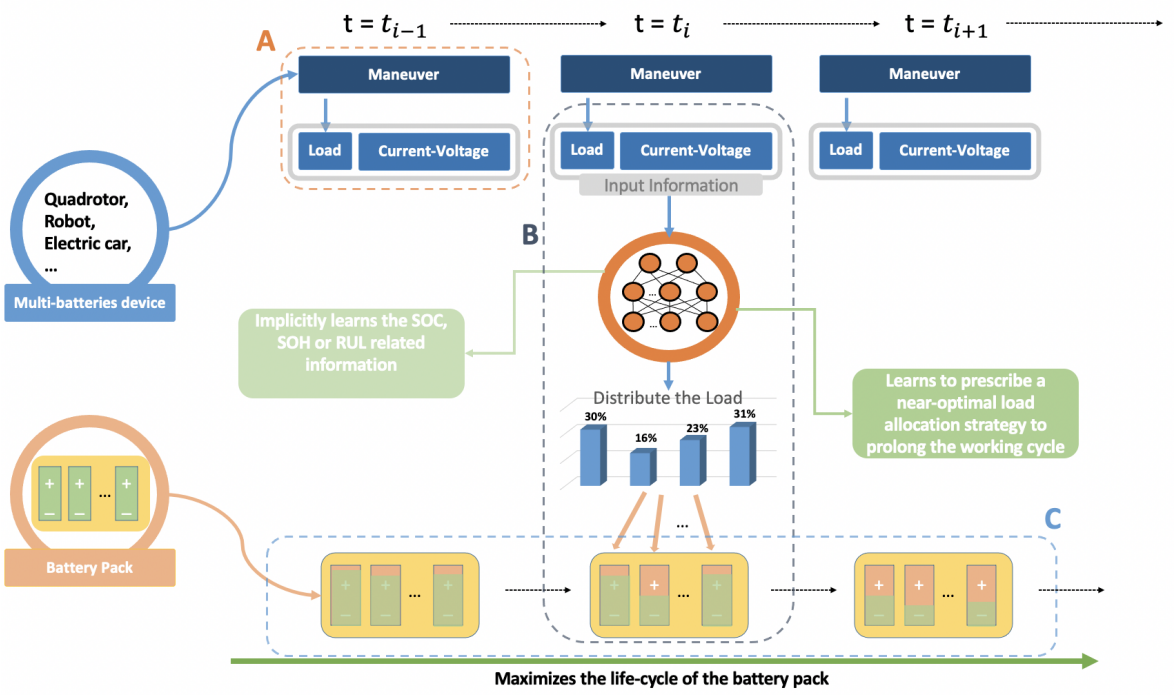

Figure 3.4: Overview of the power allocation for multi-battery systems.

output vector, and h is the output equation. For more details on the battery model, we refer interested readers to the original paper (Daigle and Kulkarni, 2013).

During the discharge process, the load is allocated at the level of a single battery cell and no load balancing is performed. This computational model serves as a reliable proxy for actual battery dynamics and allows for fast iterations over the controller design. It is worth mentioning that batteries generally have relatively complex working dynamics, which is also a challenging case study by which to evaluate the general performance of a power allocation strategy.

For training, we randomly sample battery states during operation as initial states s_0 for any new episode. The episode will be re-initialized when any of the batteries reaches the EoD state.

3.5.2 Results

The proposed framework is evaluated with respect to three performance aspects: 1) Performance is assessed on a battery system consisting of four Li-I cells. 2) Scalability is evaluated on a battery system consisting of eight Li-I cells. 3) Transferablity is evaluated on a battery system consisting of four second-life Li-I cells, where each of the batteries exhibits different degradation dynamics. 4) We compare the learning performance to other state-of-the-art reinforcement learning methods, 5) and also to heuristic strategies. We summarize the average improvement $\frac{\text{total steps(ours)-total steps(baseline)}}{\text{total steps(baseline)}}$ over the baseline; see Table. 3.2. All the performance metrics are averaged among 5000 different random initializations with random load profiles.

Experiments	Average improvement of the working cycle
four-battery system	15.2%
eight-battery system	31.9%
four-second-life-battery system	151.0%

Table 3.2: Average improvement

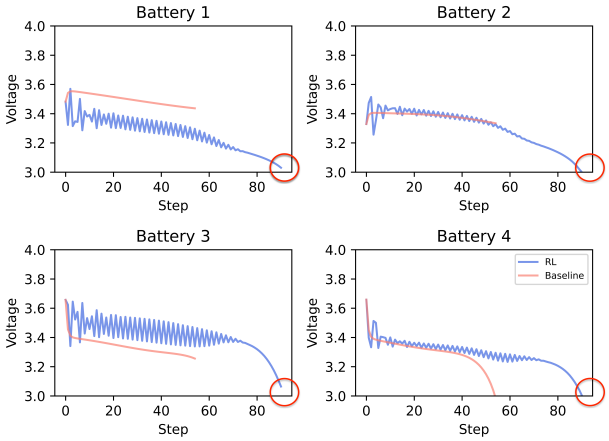

Figure 3.5: **Discharge trajectories** of a randomly selected test case: The y-axis represents the observed operational voltage of the corresponding batteries. The x-axis represents the decision-making steps.

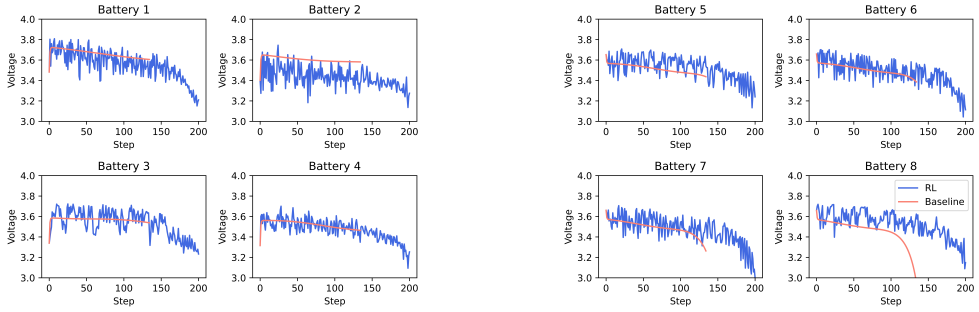

Figure 3.6: **Eight-battery system case result.** A randomly selected set of discharge trajectories from the test cases. The y-axis represents the observed operational voltage of the corresponding batteries, while the x-axis represents the decision-making steps.

1) Performance evaluation on a four-Li-I-battery system. The trained strategy is tested on 5000 different random initializations with random load profiles. Compared to the baseline strategy (distributing the power equally between all batteries), the proposed framework prolongs the working cycle by 15.2% on average. We can observe that the single batteries were controlled by the RL algorithm in such a way that they tended to reach the EoD state at approximately the same time (Fig. 3.5). This is an indication of near-optimal performance. The proposed strategy also demonstrates a relatively smooth allocation profile (Fig. 3.5).

2) Scalability evaluation on an eight-Li-I battery system. Scalability is an essential requirement for power allocation approaches since different assets will have different numbers of configurations. Previous RL approaches discretize the action and state spaces, defining different weight combinations (Xiong et al., 2018; Maia et al., 2020), which needs to redesign the action space when scaling up the system size. We present the proposed approach on an eight-battery system, following the same setup as for the system with four batteries, and show good scalability. Since more batteries provide more flexibility, the proposed RL framework again displays superior performance as compared to the baseline. The performance improvement is significantly higher when compared to the four-battery case study: Over all the test cases, the lifetime can be extended by 31.9% on average in comparison to the baseline. Similar properties can be observed as in the four-battery case: The batteries can reach the EoD state nearly simultaneously (Fig. 3.6), indicating a near-optimal allocation performance. The discharge curves are partly influenced by the allocation strategy. The oscillation represents the changing weights for the load allocated to each of the batteries. We observed

on the performed experiments that the RL policy appears to prefer frequently changing the weights between the different batteries to prolong the working cycle. The investigation of this behavior and the improvement of the interpretability will inform our future work.

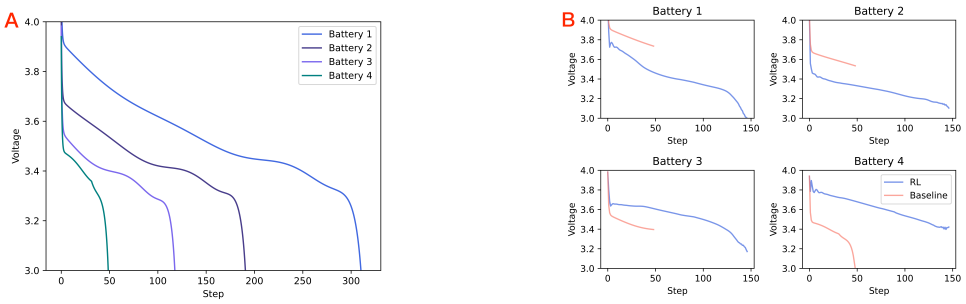

Figure 3.7: **Second-life battery system case result.** The y-axis represents the observed operational voltage of the corresponding batteries, while the x-axis represents the decision-making steps.

3) Transferablity evaluation based on a four-second-life Li-I battery system. In this research, to evaluate the transferability of the proposed approach to systems with different degradation dynamics (Chao et al., 2022), we consider batteries in second-life applications (Peterson et al., 2010; Hu et al., 2020; Fink et al., 2020). Even under the same state initialization and same load profile, second-life batteries with dissimilar degradation dynamics will reach the EoD state much earlier. In Fig. 3.7, **A** presents the voltage trajectories of four batteries. From battery 1 to 4, the degradation becomes more notable. Even under the same initialization and same load profile, the discharge curve changes significantly. **B** is a randomly chosen trajectory. The policy could significantly prolong the working cycle of the deployed second-life battery cases.

For this evaluation, we keep all settings similar to those from the previous two experiments. On average, the proposed approach achieves a 151.0% improvement as compared to the equal load distribution.

The proposed approach demonstrates even more potential in systems with different power source dynamics or degraded assets.

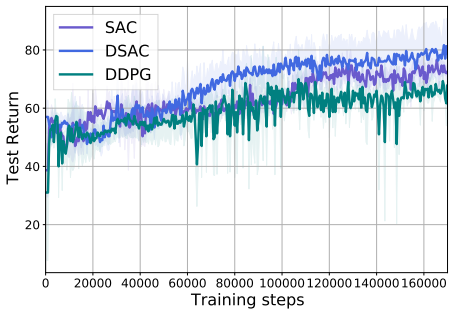

Figure 3.8: **Learning performance and reproducibility**, where the shaded areas show the standard deviation confidence intervals over three random seeds. The x-axis indicates the total time steps. The y-axis indicates the test return.

4) Learning performance compared to the state of the art (SOTA). To further evaluate the performance and reproducibility of the results of the proposed Dirichlet policy, we compare it to two alternative RL algorithms: the original SAC (Haarnoja et al., 2018a), one of the state-of-the art reinforcement learning algorithms, and the deep deterministic policy gradient (DDPG) (Lillicrap et al., 2015). We train all the agents over three different seeds on the four-battery case study. As shown in Fig. 3.8, we observe that the proposed Dirichlet-SAC (DSAC)

exhibits considerable reproducibility along with superior performance and convergence speed compared to the original SAC and the DDPG.

5) Comparison to heuristic strategies To the best of our knowledge, there are no optimization-based approaches allowing solely on voltage-current measurements. We perform this comparison for the sake of completeness of the evaluations. We would also like to emphasize that this is not a fair comparison. Since we would like to prolong the time to the EoD state, we define four heuristic strategies based on the operation voltage. We compare the average relative performance $\frac{\text{working cycle}}{\text{baseline working cycle}}$ to the baseline among 5000 different random initializations with random load profiles. However, these strategies actually yield inferior performance as compared to the baseline, see Table. 3.3.

Approaches	Weights	Relative Performance
Rule I	[0.15, 0.25, 0.25, 0.35]	0.758
Rule II	[0.1, 0.2, 0.3, 0.4]	0.279
Rule III	[0.1, 0.2, 0.2, 0.5]	0.076
Rule IV	[0.05, 0.2, 0.35, 0.45]	0.120
Proposed method	Learned	1.152

Table 3.3: Performance comparison to heuristic rules

The weights in the Table. 3.3 means to distribute the weighted load to the batteries with voltages from low to high, respectively.

3.6 Conclusion

In this work, a novel prescriptive Dirichlet policy reinforcement learning framework is proposed for continuous allocation tasks. The proposed method overcomes the bias estimation and large variance problems in policy gradient and can be applied to any general real-world allocation task. It is also compatible with all other continuous control reinforcement learning algorithms with stochastic policies. In addition, for a specific real-world prescriptive operation task, the power allocation task, we introduce the Dirichlet power allocation policy, which presents an effective and data-based prescriptive framework that is fully autonomous, flexible, transferable, and scalable. The developed framework has the potential to improve the efficiency and sustainability of multi-power source systems. To the best of our knowledge, it is also the first framework that enables distribution of the load in an end-to-end learning setup, without any additional inputs of, e.g., SoC estimation. In future work, we aim to apply and deploy the proposed framework to more challenging and extensive real-world power allocation tasks and extend it for larger problems in order to evaluate its limitations.

Acknowledgement

This work was supported by the Swiss National Science Foundation under Grant PP00P2_176878.

4 Multi-agent coordination in mixed cooperative-competitive environments

This chapter corresponds to the published article:[1]

Tian, Yuan, Klaus-Rudolf Kladny, Qin Wang, Zhiwu Huang, and Olga Fink (2023). "Multi-agent actor-critic with time dynamical opponent model". In: *Neurocomputing* 517, pp. 165–172.

Abstract: In multi-agent reinforcement learning, multiple agents learn simultaneously while interacting with a common environment and each other. Since the agents adapt their policies during learning, not only the behavior of a single agent becomes non-stationary, but also the environment as perceived by the agent. This renders it particularly challenging to perform policy improvement. In this paper, we propose to exploit the fact that the agents seek to improve their expected cumulative reward and introduce a novel *Time Dynamical Opponent Model* (TDOM) to encode the knowledge that the opponent policies tend to improve over time. We motivate TDOM theoretically by deriving a lower bound of the log objective of an individual agent and further propose *Multi-Agent Actor-Critic with Time Dynamical Opponent Model* (TDOM-AC). We evaluate the proposed TDOM-AC on a differential game and the Multi-agent Particle Environment. We show empirically that TDOM achieves superior opponent behavior prediction during test time. The proposed TDOM-AC methodology outperforms state-of-the-art Actor-Critic methods on the performed experiments in cooperative and **especially** in mixed cooperative-competitive environments. TDOM-AC results in a more stable training and a faster convergence. Our code is available at https://github.com/Yuantian013/TDOM-AC

4.1 Introduction

Multi-agent systems have recently found applications in many different domains, including traffic control (Du et al., 2021), games (Vinyals et al., 2019; Brown and Sandholm, 2019; OpenAI, 2018), consensus tracking control (Yin et al., 2022; Yuan et al., 2022) and swarm control (Hüttenrauch et al., 2019). The complexity of the tasks in these applications often precludes the usage of predefined agent behaviors and stipulates the agents to learn a policy, and to define the problem as multi-agent reinforcement learning (MARL). In such cases, multiple agents learn simultaneously while interacting with a common environment. Since the agents adapt their policies during learning, not only the behavior of a single agent becomes non-stationary, but also the environment as perceived by the agents (Hernandez-Leal et al., 2017). Since most of the conventional Reinforcement Learning (RL) approaches assume stationary system dynamics (Sutton et al., 1992), they usually perform poorly when required to interact with multiple adaptive agents in a shared environment (Lowe et al., 2017; Hernandez-Leal et al., 2017).

A common approach in MARL is to explicitly consider the presence of opponents by modeling their policies using an opponent model (Brown, 1951; Tian et al., 2019) (In the fol-

[1]Please note, this is the author's version of the manuscript published in *Neurocomputing 517 (2023): 165-172.*. Changes resulting from the publishing process, namely editing, corrections, final formatting for printed or online publication, and other modifications resulting from quality control procedures may have been subsequently added. The final publication is available at https://doi.org/10.1016/j.neucom.2022.10.045.

lowing, the word "opponents" refers to other agents in an environment irrespective of the environment's cooperative or adversarial nature). An accurate opponent model can provide informative cues to future behaviors of the opponents. However, such a precise prediction is challenging as the opponents' policies are changing over time (Tian et al., 2019).

In our novel approach, entitled *Time Dynamical Opponent Model* (TDOM), we aim to address the challenge of non-stationarity of the agent's behavior by modeling the opponent policy parameters as a dynamical system which are generally used to model the evolution of systems in time (Strogatz, 2018). Here, we build the system dynamics on the prior knowledge that all agents are concurrently trying to improve their policies with respect to their individual cumulative reward. It is worth mentioning that TDOM is highly general and can further support all kinds of opponent objectives, i.e. cooperative, competitive or mixed settings.

By deriving a lower bound on the log-objective of an individual agent, we further propose a Multi-agent Actor-Critic with Time Dynamical Opponent Model (TDOM-AC) for mixed cooperative-competitive tasks. The proposed TDOM-AC framework comprises a *Centralized Training and Decentralized Execution* (CTDE), see Fig 4.1. In this framework, centralized critics provide additional information to guide the training (Foerster et al., 2016; Lowe et al., 2017). However, this information is not used at execution time. Each agent only has access to the state information and can only select an action based on its own prediction of other opponents' actions.

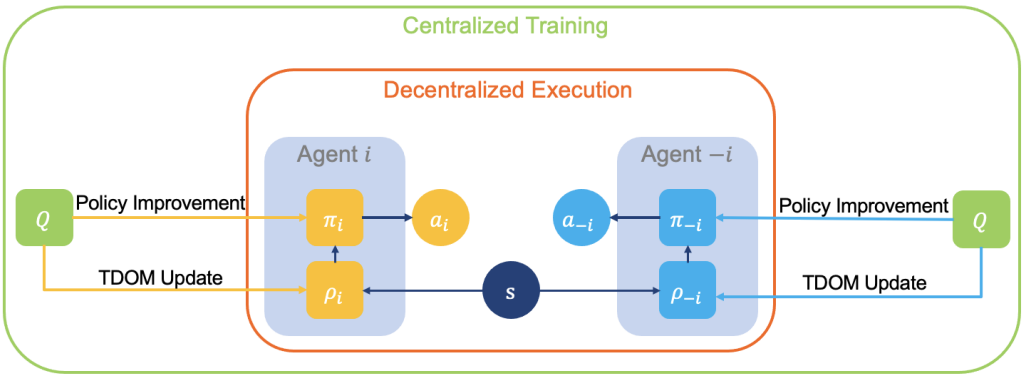

Figure 4.1: An overview of the proposed framework, where i indicates to one of the agents and $-i$ refers to other agent(s). In the proposed framework, the decision making process is fully decentralized, each agent only observes the state s, and then infers the opponent behavior(s) $\hat{a}_{-i}$ via its own opponent model $\phi_i(s)$. Based the state and predicted opponent behavior(s), the agent selects the action a_i via its policy $\pi_i(s, \hat{a}_{-i})$.

We evaluate the proposed TDOM-AC on a Differential Game and a Multi-agent Particle Environment and compare the performance to two state-of-the-art actor-critic algorithms, namely *Regularized Opponent Model with Maximum Entropy Objective* (ROMMEO) (Tian et al., 2019) and *Probabilistic Recursive Reasoning* (PR2) (Wen et al., 2019). We demonstrate empirically that the proposed TDOM algorithm achieves superior opponent behavior prediction during execution time. The proposed TDOM-AC outperforms the considered baselines on the performed experiments and considered measures. TDOM-AC results in a more stable training, faster convergence and **especially a superior performance in mixed cooperative-competitive environments**.

The remainder of this paper is organized as follows: Section 4.2 provides a brief overview of the related works of this study. Section 4.3 and 4.4 introduces the proposed opponent model and TDOM-AC. Section 4.5 interprets and compares the results of the experiments. In Section 4.6, the conclusion and future work are presented.

4.2 Related work

Multi-Agent systems (MAS) encompass decision-making of multiple agents interacting in a shared environment (Kamdar et al., 2018). For complex tasks where using predefined agent behaviors is not possible, MARL enables the agent to learn from the interaction with the environment (Zhang et al., 2021a). One of the main challenges in MARL is the inherent non-stationarity. To address this challenge, one direction has been to account for the behaviors of other agents through a centralized critic by adopting the CTDE framework (Foerster et al., 2018; Yang et al., 2018c). For value-based approaches in the CTDE framework, methods usually rely on restrictive structural constraints or network architectures, such as QDPP (Yang et al., 2020), QMIX (Rashid et al., 2018), FOP (Zhang et al., 2021c), QTRAN (Son et al., 2019), and VDN (Sunehag et al., 2017). For actor-critic based methods, these approaches usually include an additional policy with supplementary opponent models that can reason about other agents' believes (Wen et al., 2019), private information (Tian et al., 2020b), behavior (Lowe et al., 2017), strategy (Zheng et al., 2018) and other characteristics. With the supplementary opponent models, these works can also be linked to the field of opponent modeling (OM) (Albrecht and Stone, 2018; Brown, 1951).

There are several ways to model the behavior of opponents. One of them is to factorize the joint policy $\pi(a^{-i}, \mathbf{a}^{-i}|s)$ in different ways. This has been done in previous works (Brown, 1951; Tian et al., 2019; Wen et al., 2019). Also, different objective functions for the opponent model have been implemented. *Multi-agent Deep Deterministic Policy Gradient* (MADDPG) (Lowe et al., 2017) approximates opponents' policy by maximizing the log probability of other agents' actions with an entropy regularizer; PR2 (Wen et al., 2019) considers an optimization-based approximation to infer the unobservable opponent policy via variational inference (Jordan et al., 1999) and ROMMEO adopts the regularized opponent model with maximum entropy objective, which can be interpreted as a combination of MADDPG and PR2. However, the existing approaches either suffer from high computational cost due to the recursive reasoning policy gradient (Wen et al., 2019), or are limited to specific types of environments (Tian et al., 2019). In this work, we propose an alternative opponent model motivated by a temporal improvement assumption to overcome these limitations.

An earlier approach that explicitly addresses opponent-learning awareness is *Learning with Opponent Learning Awareness* (LOLA) (Foerster et al., 2017). When performing the policy update, any agent optimises its return under a one-step-look-ahead of the opponent learning. However, it is limited by strong assumptions. Specifically, these subsume access to both exact gradients and Hessians of the value function. Furthermore, a specific network design is required. Although the authors have subsequently proposed a variant of their approach, the *policy gradient-based naive learner* (NL-PG) with fewer assumptions, the intrinsic on-policy design inherently suffers from data inefficiency. Also, LOLA only supports two-agent systems, while we are considering approaches that allow for arbitrarily many agents.

4.3 Method

4.3.1 Assumptions

In this work, we aim to tackle the mentioned limitations outlined in Section 4.2. For fair comparison, we adopt the same observability assumptions from previous work (Wen et al., 2019; Tian et al., 2019; Lowe et al., 2017). Since in cooperative games all the agents receive the same reward and in zero-sum games the opponents' rewards can easily be inferred from ones own reward, we assume all agents can access each other's rewards, just like in LOLA (Foerster et al., 2017). In contrast to vanilla LOLA, we do not make the assumption of the observability of opponent policies.

4.3.2 Markov game

An N-agents Markov game (Littman, 1994), also referred to as N-agents stochastic game (Shapley, 1953), is defined by a tuple $(\mathcal{S}, \mathcal{A}^1 ... \mathcal{A}^n, r^1 ... r^n, p, \mathcal{T}, \gamma)$, where $\mathcal{S}$ is the state space, and $\mathcal{A}^i$ is the action space. At time step t, agent i chooses its action $a_t^i \in \mathcal{A}^i$ according to the policy conditioning on the observed state $s_t \in \mathcal{S}$. And $r_t^i \in \mathbb{R}$ is the corresponding rewards assigned to agent i, which is obtained from the pre-defined reward function $r_t^i = r_t^i(s_t, a_t^i, \mathbf{a}_t^{-i})$, where the $\mathbf{a}_t^{-i}$ refers to the set of opponent actions. $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is the state transition function, p is the initial state distribution and γ is the discount factor. At each time step t, actions are taken simultaneously by all agents. Each agent aims to maximize its own expected discounted sum of rewards. Thus, for each individual agent i, the objective for its policy π_i can be expressed as:

$$J(\pi_i) = \max_{\pi_i} \sum_{t=0}^{\infty} \mathbb{E}[\gamma^t r^i(s_t, a_t^i, \mathbf{a}_t^{-i})] \tag{4.1}$$

We note that since multiple adaptive agents interact in a shared environment, each agent's rewards and the environment transitions depend also on the actions of the opponents (Hernandez-Leal et al., 2017). Thus, the unobservable dynamic policies of the opponents induce non-stationarity in the environment dynamics from the perspective of a single agent. To address this challenge, we propose to consider the agent policy parameters as a dynamical system in which we encode the prior knowledge that all agents are concurrently trying to improve their policies.

4.3.3 Time dynamical opponent model

To introduce our methodology, we begin by deriving a lower bound for the maximization objective in Equation 4.1, in which we omit some of the parameterization notation for less cluttering:

$$\max_{\pi^i,\, \rho^i} \mathbb{E}_{a_t^i \sim \pi^i(\cdot|\hat{\mathbf{a}}_t^{-i}),\, \hat{\mathbf{a}}_t^{-i} \sim \rho^i,\, \mathbf{a}_t^{-i} \sim \tilde{\pi}^{-i}} \left[Q^i(s_t, a_t^i, \mathbf{a}_t^{-i}) \right], \tag{4.2}$$

where for lighter notation we omit the $s_t \sim d^\pi$, which means sampling a state from the discounted state visitation distribution d^π using current policies $\pi := \{\pi^j\}_j$, where $\pi^j(\cdot|\mathbf{a}^{-j}, s)$. $\rho^i(\cdot|s)$ refers to the belief of agent i about opponents $-i$, also known as *opponent model*. Furthermore, we define $\tilde{\pi}^j(\cdot|s)$ to be

$$\tilde{\pi}^j(\mathbf{a}^j|s) := \int_{\mathcal{A}^{-j}} \pi^j(a^j|\mathbf{a}^{-j}, s)\, \rho^j(\mathbf{a}^{-j}|s)\, d\mathbf{a}^{-j}, \tag{4.3}$$

which can be interpreted as the marginal policy of agent j. Then we can formulate the marginal opponent policies to be $\tilde{\pi}^{-i} := \{\tilde{\pi}^j\}_{j \in -i}$

The presented maximization objective means that agent i aims to maximize its Q-function given that all agents play their current policies $\tilde{\pi}_t^{-i}$ which are unknown to agent i.

We can now derive a lower bound of the log objective of agent i:

$$
\begin{aligned}
& \log \mathbb{E}_{a_t^i \sim \pi_t^i(\cdot|\hat{a}_t^{-i}),\, \hat{\mathbf{a}}_t^{-i} \sim \rho_t^i,\, \mathbf{a}_t^{-i} \sim \tilde{\pi}_t^{-i}} \left[Q^i(s_t, a_t^i,\ \mathbf{a}_t^{-i}) \right] \\
= \quad & \log \int_{\mathcal{A}^i} \int_{\mathcal{A}^{-i}} \int_{\mathcal{A}^{-i}} Q^i(s_t, a_t^i, \mathbf{a}_t^{-i})\ \tilde{\pi}_t^{-i}(\mathbf{a}_t^{-i}|s_t)\ \rho_t^i(\hat{\mathbf{a}}_t^{-i}|s_t)\ \pi_t^i(a_t^i|\hat{\mathbf{a}}_t^{-i}, s_t) \\
& d\hat{\mathbf{a}}_t^{-i}\ d\mathbf{a}_t^{-i}\ da_t^i \\
= \quad & \log \int_{\mathcal{A}^i} \int_{\mathcal{A}^{-i}} \int_{\mathcal{A}^{-i}} Q^i(s_t, a_t^i, \mathbf{a}_t^{-i})\ \frac{\tilde{\pi}_t^{-i}(\mathbf{a}_t^{-i}|s_t)}{\rho_t^i(\mathbf{a}_t^{-i}|s_t)}\ \rho_t^i(\mathbf{a}_t^{-i}|s_t)\ \rho_t^i(\hat{\mathbf{a}}_t^{-i}|s_t) \\
& \pi_t^i(a_t^i|\hat{\mathbf{a}}_t^{-i}, s_t)\ d\hat{\mathbf{a}}_t^{-i}\ d\mathbf{a}_t^{-i}\ da_t^i \\
\geq \quad & \mathbb{E}_{a_t^i \sim \pi_t^i(\cdot|\hat{\mathbf{a}}_t^{-i}),\, \hat{\mathbf{a}}_t^{-i} \sim \rho_t^i,\, \mathbf{a}_t^{-i} \sim \rho_t^i} \left[\log Q^i(s_t, a_t^i, \mathbf{a}_t^{-i}) \ +\ \log \left(\frac{\tilde{\pi}^{-i}(\mathbf{a}_t^{-i}|s_t)}{\rho_t^i(\mathbf{a}_t^{-i}|s_t)} \right) \right] \\
= \quad & \mathbb{E}_{a_t^i \sim \tilde{\pi}_t^i,\, \mathbf{a}_t^{-i} \sim \rho_t^i} \left[\log Q^i(s_t, a_t^i, \mathbf{a}_t^{-i}) \right] \ -\ \mathrm{KL}\left(\rho_t^i(\cdot|s_t) \,||\, \tilde{\pi}_t^{-i}(\cdot|s_t) \right).
\end{aligned}
\tag{4.4}
$$

If we furthermore make the assumption that:

$$
Q^{\text{opt}} = \max_{a^i} Q^i(s_t, a^i, \mathbf{a}^{-i}) \quad \forall \mathbf{a}^{-i} \in \mathcal{A}^{-i}, \tag{4.5}
$$

for some fixed Q^{opt}, we see that we can maximize this lower bound by minimizing the Kullback-Leibler Divergence $\mathrm{KL}\left(\rho_t^i(\cdot|s) \,||\, \tilde{\pi}_t^{-i}(\cdot|s_t) \right)$ w.r.t. ρ_t^i and then maximizing the Q-function w.r.t. π_t^i:

$$
\max_{\pi_t^i} \mathbb{E}_{a_t^i \sim \pi_t^i(\cdot|\mathbf{a}_t^{-i}),\, \mathbf{a}_t^{-i} \sim \rho_t^i} \left[\ Q^i(s_t, a_t^i, \mathbf{a}_t^{-i}) \right]. \tag{4.6}
$$

However, the method proposed above has an obvious issue: How can we minimize $\mathrm{KL}\left(\rho_t^i(\cdot|s) \,||\, \tilde{\pi}_t^{-i}(\cdot|s_t) \right)$ if $\tilde{\pi}_t^{-i}$ is not available to agent i?

In order to address this question, we propose to utilize prior information about the opponents' learning process. Using this information would enable to better model their non-stationary behavior. Specifically, there exists one aspect that to the best of our knowledge has not been considered before in opponent modelling: Over time, each agent j is expected to improve its policy using policy network parameters θ_t^j [2] in order to maximize its expected cumulative reward under the given system dynamics and opponent policies. This can be expressed as an ordinary differential equation (ODE):

$$
\begin{aligned}
\frac{d}{dt}\, \theta_t^j \ &\approx\ \nabla_{\theta^j}\, \mathbb{E}_{\pi_{\theta_t}^j,\, \pi_t^{-j}} \left[\sum_{t=0}^{\infty} \gamma^t\ r_t^j(a_t^j,\ \mathbf{a}_t^{-j},\ s_t) \right] \\
&=\ \nabla_{\theta^j}\, \mathbb{E}_{\pi_\theta^j,\, \pi_t^{-j}} \left[Q^j(a_t^j,\ \mathbf{a}_t^{-j},\ s_t) \right],
\end{aligned}
\tag{4.7}
$$

where $\pi^{-j} := \{\pi^k\}_{k \in -j}$.

We propose to encode this knowledge in the opponent model design. We would like to make explicit here that unlike in policy improvement, the opponent model is designed to simulate the policy optimization process for all opponents instead of maximizing their expected Q-value. It is worth to point out that unlike LOLA (Foerster et al., 2017) which considers the opponent's policy update to optimize the agent's policy, our agent takes the opponents'

[2] When parameterizing a function for agent j, we will always write e.g. π_θ^j instead of $\pi_{\theta^j}^j$.

policy improvement assumption into account to optimize its opponent model instead of the policy directly.

In order to minimize KL $\left(\rho^i(\cdot|s) \,||\, \tilde{\pi}^{-i}(\cdot|s_t)\right)$, we exploit the temporal improvement assumption for discrete time dynamics, parameterized by θ^{-i}:

$$\theta_t^{-i} \;\approx\; \theta_{t-1}^{-i} \;+\; \eta \nabla_{\theta^{-i}} \, \mathbb{E}_{\tilde{\pi}_{t-1}^i,\, \tilde{\pi}_{\theta}^{-i}} \left[Q^{-i}(a^i,\, \mathbf{a}^{-i},\, s) \right], \tag{4.8}$$

for some $\eta > 0$. However, the opponent model cannot be updated like this since neither $\tilde{\pi}_{\theta_t}^{-i}$ nor θ_t^{-i} are directly available to agent i. Hence, we take our best approximation $\rho_{\psi_t}^i$ which is our opponent model which is parameterized by ψ_t^i and update as

$$\psi_t^i \;\leftarrow\; \psi_{t-1}^i \;+\; \eta \nabla_{\psi^i} \, \mathbb{E}_{a^i \sim \tilde{\pi}_{t-1}^i,\, \mathbf{a}^{-i} \sim \rho_{\psi_t^i}} \left[Q^{-i}(a^i,\, \mathbf{a}^{-i},\, s) \right]. \tag{4.9}$$

We point out that the Q mentioned above can represent any type of critic function, such as Q-function, soft Q-function or advantage function.

To summarize, firstly, we derive a learning objective for agent i's policy π^i. We show that a proper opponent model can alleviate the non-stationarity problem of policy update in MARL. With an accurate opponent prediction, each agent can access to a more reliable Q estimation, which provides a better guidance for its own policy update and further allows the agent become to a better collaborator or stronger adversary to influence other agents in cooperative and competitive setting respectively. Secondly, we propose a novel approach to exploit the temporal improvement assumption to guide the opponent model evolution.

4.4 Multi-Agent Actor-Critic with time dynamical opponent model (TDOM-AC)

With the proposed TDOM, we introduce Multi-Agent Actor-Critic with Time Dynamical Opponent Model (TDOM-AC). TDOM-AC follows the CTDE framework (Foerster et al., 2016; Lowe et al., 2017). There are three main modules in the proposed TDOM-AC: Centralized Q-function $Q(s, a^i, \mathbf{a}^{-i})$, opponent model $\rho(\cdot|s)$ and policy $\pi(\cdot|s, \hat{\mathbf{a}}^{-i})$. We further use neural networks (NNs) as function approximators, particularly applicable in high-dimensional and/or continuous multi-agent tasks. For an individual agent, i, the three modules are parameterized by ϕ^i, θ^i and ψ^i, respectively. The functions are updated using stochastic gradient based optimization with learning rates $\eta_\cdot$:

$$\begin{aligned} \phi_{t+1}^i &\leftarrow \phi_t^i + \eta_\phi \hat{\nabla}_{\phi^i} J(\phi_t^i) \\ \theta_{t+1}^i &\leftarrow \theta_t^i + \eta_\theta \hat{\nabla}_{\theta^i} J(\theta_t^i) \end{aligned} \tag{4.10}$$

and as elucidated in subsection 4.3.3,

$$\psi_{t+1}^i \;\leftarrow\; \psi_t^i + \eta_\psi \hat{\nabla}_{\psi^i} J(\psi_t^i). \tag{4.11}$$

We would like to clarify that although Equations 4.10 and 4.11 perform similar operations, their underlying idea is different. We can interpret Equation 4.10 as an approximation of a policy improvement and evaluation step without running it until convergence. However, Equation 4.11 does not follow this idea. Instead, this update is based on the temporal improvement assumption with the underlying goal of minimizing the Kullback-Leibler divergence to the true marginal opponent policies $\tilde{\pi}^{-i}$ instead of policy improvement.

In the proposed TDOM-AC, experience replay buffer D is used (Mnih et al., 2015), where the off-policy experiences of all agents are recorded. In a scenario with N agents, at time step t, a tuple $[s_t, s_{t+1}, a_t^1, ..., a_t^N, r_t^1, ..., r_t^N]$ is recorded.

We adopt the maximum entropy reinforcement learning (MERL) framework (Haarnoja et al., 2018a) to enable a richer exploration and a better learning stability. It is easy to see that the derivation still holds. We merely omit the adjustments in the previous sections for the purpose of readability. The centralized soft Q-function parameters can be trained to minimize the soft Bellman residual:

$$J(\phi^i) = \mathbb{E}_{(s_t, a_t, \mathbf{a}_t^{-i}, s_{t+1})\sim D} \frac{1}{2}[Q_\phi^i(s_t, a_t, \mathbf{a}_t^{-i}) - (r_t^i + \gamma V(s_{t+1}))]^2, \tag{4.12}$$

where the value function V is implicitly parameterized by the soft Q-function (Haarnoja et al., 2018a) parameters. The objective function becomes:

$$\begin{aligned} J(\phi^i) = & -\mathbb{E}_{(s_t, a_t, s_{t+1})\sim D,\, \hat{\mathbf{a}}_{t+1}^{-i}\sim\rho_\psi^i,\, \hat{a}_{t+1}^i\sim\pi_\theta^i} \Big[\Big(Q_\phi^i(s_t, a_t^i, \mathbf{a}_t^{-i}) - \big(r^i(s_t, a_t^i, \mathbf{a}_t^{-i}) \\ & + \gamma\big(Q_{\overline{\phi}}^i(s_{t+1}, \hat{a}_{t+1}^i, \hat{\mathbf{a}}_{t+1}^{-i}) - \alpha \log \pi^i(\hat{a}_{t+1}^i | s_{t+1}, \hat{\mathbf{a}}_{t+1}^{-i}) \\ & - \alpha \log \rho_\psi^i(\hat{\mathbf{a}}_{t+1}^{-i} | s_{t+1})\big)\big)\Big)^2\Big]. \end{aligned} \tag{4.13}$$

The $Q_{\overline{\phi}}^i$ is the target soft Q-network that has the same structure as Q^i and is parameterized by $\overline{\phi^i}$, but updated through exponentially moving average of the soft Q-function weights (Mnih et al., 2015).

According to the MERL objective, the TDOM-based policy is learned by directly minimizing the expected KL-divergence between normalized centralized soft Q-function:

$$J(\theta^i) = \mathbb{E}_{s\sim D,\, \hat{\mathbf{a}}_{t+1}^{-i}\sim\rho_\psi^i}[Q_\phi^i(s, a^i, \hat{\mathbf{a}}^{-i}) - \alpha \log \pi_\theta^i(a^i | s, \hat{\mathbf{a}}^{-i})], \tag{4.14}$$

where α is the temperature parameter that determines the relative importance of the entropy term versus the reward, thus controls the stochasticity of the optimal policy. In order to achieve a low variance estimator of $J(\theta^i)$, we apply the reparameterization trick (Kingma and Welling, 2014) for modeling the policy:

$$a_t^i = f_\theta^i(\epsilon; s, \mathbf{a}^{-i}), \tag{4.15}$$

where ϵ_t is a noise vector that is sampled from a fixed distribution. A common choice is a Gaussian distribution $\mathcal{N}$. We can now rewrite the objective in Equation 4.14 as

$$\begin{aligned} J(\theta^i) = & \mathbb{E}_{s\sim D,\, \hat{\mathbf{a}}_{t+1}^{-i}\sim\rho_\psi^i,\, \epsilon\sim\mathcal{N}}[Q^i(s, f_\theta^i(\epsilon; s, \mathbf{a}^{-i}), \mathbf{a}^{-i}) \\ & - \alpha \log \pi_\theta^i(f_\theta^i(\epsilon; s, \mathbf{a}^{-i}) | s, \mathbf{a}^{-i})]. \end{aligned} \tag{4.16}$$

Let $\mathbf{Q}^{-i}(s, a^i, \hat{\mathbf{a}}^{-i}) := \sum_{j\in -i} Q_\phi^j(s, a^i, \hat{\mathbf{a}}^{-i})$. Then, according to Equation 4.9, the objective for the TDOM model can be written as

$$J(\psi^i) = \mathbb{E}_{s\sim D,\, \hat{\mathbf{a}}^{-i}\sim\rho_\psi^i,\, a^i\sim\pi_\theta^i}[\mathbf{Q}^{-i}(s, a^i, \hat{\mathbf{a}}^{-i}) - \alpha \log \rho_\psi^i(\hat{\mathbf{a}}^{-i} | s)]. \tag{4.17}$$

However, in mixed cooperative-competitive environments, agents may have conflicting interests which can *neutralize* the gradient in this formulation. We illustrate this by an example of a two-player zero-sum Markov game:

$$r^1(s, a^1, a^2) = -r^2(s, a^1, a^2), \qquad \forall s \in \mathcal{S},\ (a^1, a^2) \in \mathcal{A}^2. \tag{4.18}$$

The Q-function approximations Q_ϕ^1 and Q_ϕ^2 for agent 1 and agent 2 respectively, have converged to their true functions $Q_{\pi_\theta^1, \pi_\theta^2}^1$ and $Q_{\pi_\theta^1, \pi_\theta^2}^2$.

Theorem 1. *In this setting, the gradient $\nabla_{\psi^i} J(\psi^i)$ is exclusively determined by entropy terms.*

Proof. With $p(\tau)$ denoting the trajectory distribution, observe that the structure of the true $Q^1_{\pi^1_\theta,\pi^2_\theta}$ is:

$$\begin{aligned}
& Q^1_{\pi^1_\theta,\pi^2_\theta}(s_0,a^1_0,a^2_0) \\
\triangleq\ & r^1(s_0,a^1_0,a^2_0)+\mathbb{E}_{s\sim p(s_1|a^1_0,a^2_0)}\Big(\gamma V^1_{\pi^1_\theta,\pi^2_\theta}(s_1)\Big) \\
\triangleq\ & r^1(s_0,a^1_0,a^2_0)+\mathbb{E}_{\tau\sim p(\tau)}\left[\sum_{t=1}^{\infty}\gamma^t\Big(r^1_t(s_t,a^1_t,a^2_t)\mathcal{H}(\pi^1_\theta(a^1_t|s_t,a^2_t)\rho^1_\psi(a^2_t|s_t))\Big)\right] \\
=\ & -r^2(s_0,a^1_0,a^2_0)-\mathbb{E}_{\tau\sim p(\tau)}\left[\sum_{t=1}^{\infty}\gamma^t\Big(r^2_t(s_t,a^1_t,a^2_t)+\mathcal{H}(\pi^1_\theta(a^1_t|s_t,a^2_t)\rho^1_\psi(a^2_t|s_t))\Big)\right] \\
=\ & \Big(\sum_{t=1}^{\infty}\gamma^t\mathcal{H}(\pi^2_\theta(a^2_t|s_t,a^1_t)\rho^2_\psi(a^1_t|s_t))-\gamma^t\mathcal{H}(\pi^1_\theta(a^1_t|s_t,a^2_t)\rho^1_\psi(a^2_t|s_t))\Big) \\
& -Q^2_{\pi^1_\theta,\pi^2_\theta}(s_0,a^1_0,a^2_0),
\end{aligned} \tag{4.19}$$

where $\mathcal{H}(\cdot)$ denotes Shannon entropy. For lighter notation, let

$$\mathcal{E}=\Big(\sum_{t=1}^{\infty}\gamma^t\mathcal{H}(\pi^2(a^2_t|s_t,a^1_t)\rho^2_\psi(a^1_t|s_t))-\gamma^t\mathcal{H}(\pi^1(a^1_t|s_t,a^2_t)\rho^1_\psi(a^2_t|s_t))\Big). \tag{4.20}$$

Then, we can determine the gradient as:

$$\begin{aligned}
\nabla_{\psi^i}J(\psi^i) &= \mathbb{E}_{\tau\sim p,\,\epsilon\sim\mathcal{N}}\big[\nabla_{\psi^i}Q^2_{\pi^1,\pi^2}(s_0,f^i_\psi(\epsilon;s_0))-\nabla_{\psi^i}Q^2_{\pi^1,\pi^2}(s_0,f^i_\psi(\epsilon;s_0)) \\
&\quad+\nabla_{\psi^i}\mathcal{E}-\alpha\nabla_{\psi^i}\log(\rho^i_\psi(f^i_\psi(\epsilon;s_0)|s_0))\big] \\
&= \mathbb{E}_{\tau\sim p,\,\epsilon\sim\mathcal{N}}\big[\nabla_{\psi^i}\mathcal{E}-\alpha\nabla_{\psi^i}\log(\rho^i_\psi(f^i_\psi(\epsilon;s_0)|s_0))\big] \\
&= \mathbb{E}_{\tau\sim p}\big[\nabla_{\psi^i}\mathcal{E}+\alpha\nabla_{\psi^i}\mathcal{H}(\rho^i_\psi(\cdot,\cdot|s_0))\big].
\end{aligned} \tag{4.21}$$

□

To alleviate the potential issue of neutralized gradients, we propose to modify the TDOM objective to be based on empirical data. Specifically, we modify the objective function as

$$J(\psi^i)\ =\mathbb{E}_{(s,\mathbf{a}^{-j})\sim D,\,\hat{a}^j\sim\rho^i_\psi}\Big[\sum_{j\in -i}Q^j_\phi\left(s,\hat{a}^j,\mathbf{a}^{-i\setminus\{j\}},a^i\right)-\alpha\log\rho^j_\psi(\hat{\mathbf{a}}^{-j}|s)\Big]. \tag{4.22}$$

Note that again we use the reparameterization trick (Haarnoja et al., 2018a) in order to be able to exchange expectation and gradient, while still sampling from the opponent model ρ^i_ψ. The pseudo-code can be found below 2.

4.5 Experiments

We compare the proposed TDOM-AC to two state-of-the-art algorithms based on opponent modelling: PR2 (Wen et al., 2019) and ROMMEO (Tian et al., 2019), which have shown a better performance with respect to the considered measures compared to *Multi-Agent Soft Q-Learning* MASQL (Wei et al., 2018) and MADDPG (Lowe et al., 2017) in previous studies. We evaluate the performance of the proposed TDOM-AC methods on a differential game (Wei et al., 2018; Wen et al., 2019; Tian et al., 2019) and the multi-agent particle environments (Lowe et al., 2017). Those tasks contain fully cooperative and mixed cooperative-competitive objectives with challenging non-trivial equilibria (Wen et al., 2019) and continuous action space. All the experiments are adopted from PR2 and ROMMEO for adequate comparison.

Initialize replay buffer D and randomly initialize N soft Q networks $Q^{1..n}_{\phi_{i..n}}$, N policy networks $\pi^{1..n}_{\theta_{1..n}}$, and opponent model $\rho^{1..n}_{\psi_{1..n}}$ with parameters $\phi_{i..n}$, $\theta_{1..n}$ and $\psi_{1..n}$.
Initialize the parameters of target networks with $Q^{1..n}_{\bar{\phi}_{1..n}}$
for each iteration **do**
  Sample s_0 according to $p_0(\cdot)$
  while Not done **do**
    for each agent **do**
      Sample $\hat{\mathbf{a}}_t^{-i}$ from $\rho^i(\cdot|s_t)$ and a_t^i from $\pi^i(\cdot|s_t, \hat{\mathbf{a}}_t^{-i})$
      Combine the true actions $\mathbf{a}_t = [a_t^1, ..., a_t^n]$ and take one step forward
    end for
    Observe s_{t+1}, $\mathbf{r}_t = [r_t^1, ..., r_t^n]$ and store $(s_t, \mathbf{a}_t, \mathbf{r}_t, s_{t+1})$ in D
    Sample minibatches of N transitions from D
    for each agent **do**
      Estimate policy gradient according to Equations 4.13,4.16, and 4.22:

$$\begin{aligned}\phi^i_{t+1} &\leftarrow \phi^i_t + \eta_\phi \hat{\nabla}_{\phi^i} J(\phi^i_t)\\ \theta^i_{t+1} &\leftarrow \theta^i_t + \eta_\theta \hat{\nabla}_{\theta^i} J(\theta^i_t)\\ \psi^i_{t+1} &\leftarrow \psi^i_t + \eta_\psi \hat{\nabla}_{\psi^i} J(\psi^i_t).\end{aligned}$$

      Update the parameters of target networks $Q^{1..n}_{\bar{\phi}_{1..n}}$
    end for
  end while
end for

Algorithm 2: Multi-agent Actor-Critic with Time Dynamical Opponent Model (TDOM-AC)

To reduce the performance difference caused solely by entropy regularization, we add an entropy term to the PR2 objective and equip it with a stochastic policy since both TDOM-AC and ROMMEO employ the maximum entropy reinforcement learning framework. This has been shown to yield better exploration and sample efficiency (Haarnoja et al., 2018a).

For the experiment settings, all policies and opponent models use a fully connected multi-layer perceptron (MLP) with two hidden layers of 256 units each, outputting the mean μ and standard deviation σ of a univariate Gaussian distribution. All hidden layers use the leaky-RelU activation function and we adopt the same invertible squashing function technique as (Haarnoja et al., 2018a) for the output layer. For the Q-network, we use a fully-connected MLP with two hidden layers of 256 units with leaky-Relu activation function, outputting the Q-value. We employ the Adam optimizer with the learning rate $3e-4$ and batch size 256. The target smoothing coefficient τ, entropy control parameter α and the discount factor γ are $0.01, 1$, and 0.95 respectively. All training hyper-parameters are derived from the SAC algorithm (as published in (Haarnoja et al., 2018a)) without any additional adaptations.

4.5.1 Differential game

The differential Max-of-Two Quadratic Game is a single step continuous action space decision making task, where the gradient update tends to direct the training agent to a sub-optimal point (Tian et al., 2019). The reward surface is displayed in the Fig 4.2. There exists a local maximum 0 at $(-5,-5)$ and a global maximum 10 at $(5,5)$, with a deep valley positioned in the middle. The agents are rewarded by their joint actions, following the rule: $r_1 = r_2 = max(f_1, f_2)$, where $f_1 = 0.8*[-(\frac{a_1+5}{3})^2-(\frac{a_2+5}{3})^2]$ and $f_1 = [-(\frac{a_1-5}{1})^2-(\frac{a_2-5}{1})^2]+10$. Both of the agents have the same continuous action space in the range $[-10, 10]$. Compared

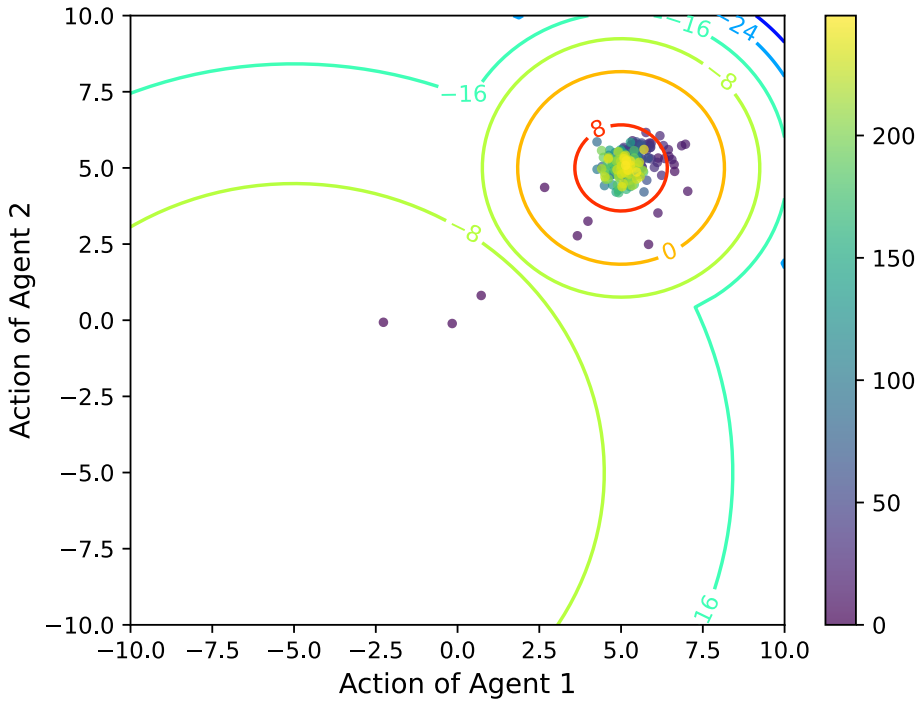

Figure 4.2: Reward surface and learning path of agents trained by TDOM-AC. Scattered points are actions taken at each step, the lighter points are sampled later during training.

to other state-of-the art approaches, TDOM-AC shows a superior performance. In Fig. 4.3, the learning path of the proposed TDOM-AC is displayed, where the lighter (yellow) dots

Methods	TDOM-AC	ROMMEO	PR2
Running time	**0.068s**	0.089s	0.436s

Table 4.1: Average running time (seconds) per update of different methods

are sampled later. This indicates a stable and fast convergence. In Fig. 4.3, the learning curves of all considered algorithms are displayed. Both TDOM-AC and ROMMEO show a fast and stable convergence. However, ROMMEO fails for some random seeds, resulting in a lower average performance. We note that the maximum-entropy version of PR2 indeed converges faster than the original version (Wen et al., 2019). Nevertheless, the learning process fluctuates significantly and it suffers from substantial computational cost, see Table 4.1.

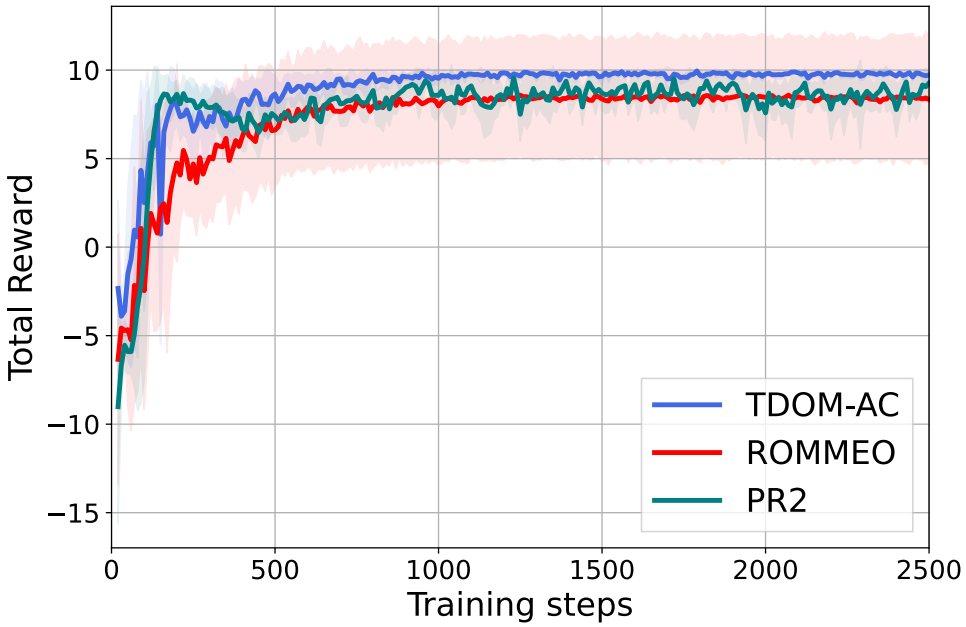

Figure 4.3: Average performance of TDOM-AC and other baselines, where the shaded areas show the 1-SD confidence intervals over multiple random seeds

4.5.2 Cooperative navigation

Cooperative Navigation is a three-agent fully cooperative task. The three agents should learn to cooperate to reach and cover three randomly generated landmarks. The agents can observe the relative positions of other agents and landmarks and are collectively rewarded based on the proximity of any agent to each landmark. Besides this, the agents are being penalized when colliding with each other. The expected behavior is to "cover" the three landmarks as fast as possible without any collision. The result shows that TDOM-AC outperforms all other considered baseline algorithms in terms of both faster convergence and a better performance, see Fig 4.4. Also, the TDOM-AC attains more accurate opponent behavior prediction, despite the fact that the agents do not have direct access to any opponent action distribution, see Fig 4.5. This is in contrast to ROMMEO, which utilizes a regularized opponent model, the regularization being the KL divergence between the opponent model and the empirical opponent distribution.

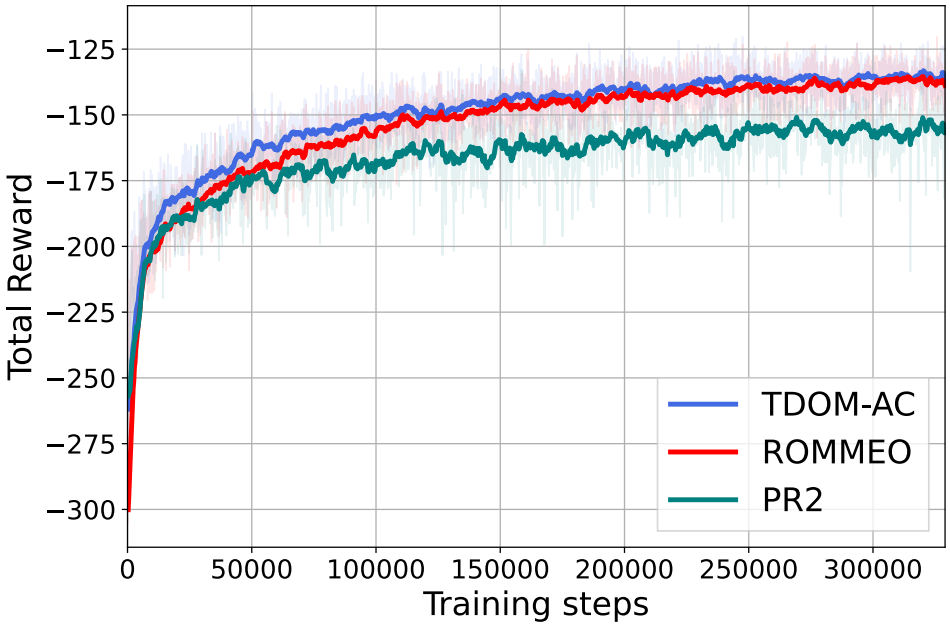

Figure 4.4: Moving average of total reward of TDOM-AC and other baselines on Cooperative Navigation.

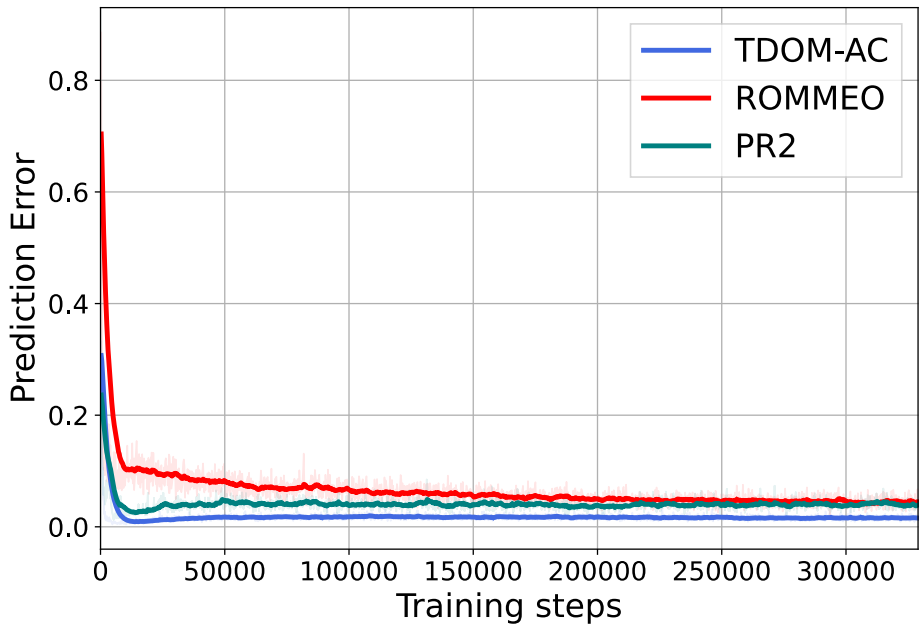

Figure 4.5: The test time opponents' behaviors prediction error of TDOM-AC and other baselines

4.5.3 Predator and prey

Predator and Prey is a challenging four-agent mixed cooperative-competitive task. There are three slower cooperating adversaries that try to chase the faster agent in a randomly generated environment with two large landmarks impeding the way. The cooperative adversaries are rewarded for every collision with the agent, while the agent is being penalized for any such collision. All agents can observe the relative positions and velocities of other agents and the positions of the landmarks.

For this task, we train all the algorithms for 0.6M steps and compare the normalized average episode advantage score (the sum of agent's rewards in an episode - the sum of adversaries' rewards in an episode (Wen et al., 2019; Lowe et al., 2017). We evaluate the performance of the different algorithms by letting the cooperative adversaries trained by one algorithm play against an agent trained by another algorithm and vice versa. A higher score means the agent (prey) performs better than the cooperative adversaries (predators), while a lower score means that the cooperative adversaries have a superior policy over the agent. Table 4.2 shows that the TDOM-AC performs best on both prey (0.999) and predator (0.547) side.

Ag vs. Ads	TDOM-AC	ROMMEO	PR2	Mean
TDOM-AC	0.967	1.000	0.999	**0.989**
ROMMEO	0.674	0.997	0.981	0.884
PR2	0.000	0.722	0.313	0.345
Mean	**0.547**	0.906	0.764	N/A

Table 4.2: Comparison of different model settings (Agent vs. Adversaries). The values are the normalized average episode advantage scores.

4.6 Conclusion

In this work, we propose a novel time dynamical opponent model called TDOM. It supports mixed cooperative-competitive tasks with a low computational cost. Furthermore, we introduce the TDOM-AC algorithm and demonstrate the superior performance compared to other state-of-the-art methods on multiple challenging benchmarks. In the future, we plan to omit the centralized training and instead also model opponent Q-function parameters as time dynamical latent variables, thereby relying exclusively on past opponent actions for training. Also, we would like to evaluate the proposed approach on partially observable environments where the agent does not share its observation space with all opponents.

5 Effective and efficient black-box optimization via reinforcement learning

This chapter corresponds to the published article:[1]

Tian, Yuan, Qin Wang, Zhiwu Huang, Wen Li, Dengxin Dai, Minghao Yang, Jun Wang, and Olga Fink (2020a). "Off-policy reinforcement learning for efficient and effective gan architecture search". In: *European Conference on Computer Vision*. Springer, pp. 175–192.

Abstract: In this paper, we introduce a new reinforcement learning (RL) based neural architecture search (NAS) methodology for effective and efficient generative adversarial network (GAN) architecture search. The key idea is to formulate the GAN architecture search problem as a Markov decision process (MDP) for smoother architecture sampling, which enables a more effective RL-based search algorithm by targeting the potential global optimal architecture. To improve efficiency, we exploit an off-policy GAN architecture search algorithm that makes efficient use of the samples generated by previous policies. Evaluation on two standard benchmark datasets (i.e., CIFAR-10 and STL-10) demonstrates that the proposed method is able to discover highly competitive architectures for generally better image generation results with a considerably reduced computational burden: 7 GPU hours. Our code is available at https://github.com/Yuantian013/E2GAN.

5.1 Introduction

Generative adversarial networks (GANs) have been successfully applied to a wide range of generation tasks, including image generation (Goodfellow et al., 2014; Brock et al., 2018; Bao et al., 2017; Wang et al., 2018; Guo et al., 2019a), text to image synthesis (Reed et al., 2016; Zhang et al., 2017; Park et al., 2019) and image translation (Isola et al., 2017; Choi et al., 2018), to name a few. To further improve the generation quality, several extensions and further developments have been proposed, ranging from regularization terms (Brock et al., 2016; Gulrajani et al., 2017), progressive training strategy (Karras et al., 2017), utilizing attention mechanism (Xu et al., 2018; Zhang et al., 2019), and to new architectures (Karras et al., 2019; Brock et al., 2018).

While designing favorable neural architectures of GANs has made great success, it typically requires a large amount of time, effort, and domain expertise. For instance, several state-of-the-art GANs (Karras et al., 2019; Brock et al., 2018) design appreciably complex generator or discriminator backbones for better generating high-resolution images. To alleviate the network engineering pain, an efficient automated architecture searching framework for GAN is highly needed. On the other hand, Neural architecture search (NAS) has been applied and proved effective in discriminative tasks such as image classification (Krizhevsky et al., 2012) and segmentation (Liu et al., 2019b). Encouraged by this, AGAN (Wang and Huan, 2019) and AutoGAN (Gong et al., 2019) have introduced neural architecture search methods

[1] Please note, this is the author's version of the manuscript published in *European Conference on Computer Vision. Springer, Cham, 2020: 175-192.*. Changes resulting from the publishing process, namely editing, corrections, final formatting for printed or online publication, and other modifications resulting from quality control procedures may have been subsequently added. The final publication is available at https://doi.org/10.1007/978-3-030-58571-6_11.

for GAN based on reinforcement learning (RL), thereby enabling a significant speedup of architecture searching process.

Similar to the other architecture search tasks (image classification, image segmentation), recently proposed RL-based GAN architecture search method AGAN (Wang and Huan, 2019) optimized the entire architecture. Since the same policy might sample different architectures, it is likely to suffer from noisy gradients and a high variance, which potentially further harms the policy update stability. To circumvent this issue, multi-level architecture search (MLAS) has been used in AutoGAN (Wang and Huan, 2019), and a progressive optimization formulation is used. However, because optimization is based on the best performance of the current architecture level, this progressive formulation potentially leads to a local minimum solution.

To overcome these drawbacks, we reformulate the GAN architecture search problem as a Markov decision process (MDP). The new formulation is partly inspired by the human-designed Progressive GAN (Karras et al., 2017), which has shown to improve generation quality progressively in intermediate outputs of each architecture cell. In our new formulation, a sequence of decisions will be made during the entire architecture design process, which allows state-based sampling and thus alleviates the variance. In addition, as we will show later in the paper, by using a carefully designed reward, this new formulation also allows us to target effective global optimization over the entire architecture.

More importantly, the MDP formulation can better facilitate off-policy RL training to improve data efficiency. The previously proposed RL-based GAN architecture search methods (Gong et al., 2019; Wang and Huan, 2019) are based on on-policy RL, leading to limited data efficiency that results in considerably long training time. Specifically, on-policy RL approach generally requires frequent sampling of a batch of architectures generated by current policy to update the policy. Moreover, new samples are required to be collected for each gradient step, while the previous batches are directly disposed. This quickly becomes very expensive as the number of gradient steps and samples increases with the complexity of the task, especially in the architecture search tasks. by comparison, off-policy reinforcement learning algorithms make use of past experience such that the RL agents are enabled to learn more efficiently. This has been proven to be effective in other RL tasks, including legged locomotion (Lillicrap et al., 2015) and complex video games (Mnih et al., 2015). However, exploiting off-policy data for GAN architecture search poses new challenges. Training the policy network inevitably becomes unstable by using off-policy data, because these training samples are systematically different from the on-policy ones. This presents a great challenge to the stability and convergence of the algorithm (Bhatnagar et al., 2009). Our proposed MDP formulation can make a difference here. By allowing state-based sampling, the new formulation alleviates this instability, and better supports the off-policy strategy.

The contributions of this paper are two-fold:

1. We reformulate the problem of neural architecture search for GAN as an MDP for smoother architecture sampling, which enables a more effective RL-based search algorithm and potentially more global optimization.

2. We propose an efficient and effective off-policy RL NAS framework for GAN architecture search (E^2GAN), which is 6 times faster than existing RL-based GAN search approaches with competitive performance.

We conduct a variety of experiments to validate the effectiveness of E^2GAN. Our discovered architectures yield better results compared to RL-based competitors. E^2GAN is efficient, as it is able to to find a highly competitive model within **7 GPU hours**.

5.2 Related work

Reinforcement learning Recent progress in model-free reinforcement learning (RL) (Sutton et al., 1992) has fostered promising results in many interesting tasks ranging from gaming (Mnih et al., 2013; Silver et al., 2014), to planning and control problems (Hwangbo et al., 2019; Kumar et al., 2016; Xie et al., 2019; Han et al., 2019a; Chao et al., 2020; Han et al., 2020) and even up to the AutoML (Zoph and Le, 2017; Pham et al., 2018; Liu et al., 2018). However, model-free deep RL methods are notoriously expensive in terms of their sample complexity. One reason of the poor sample efficiency is the use of on-policy reinforcement learning algorithms, such as trust region policy optimization (TRPO) (Schulman et al., 2015), proximal policy optimization(PPO) (Schulman et al., 2017) or REINFORCE (Williams, 1992). On-policy learning algorithms **require new samples generated by the current policy for each gradient step**. On the contrary, off-policy algorithms aim to reuse past experience. Recent developments of the off-policy reinforcement learning algorithms, such as soft Actor-Critic (SAC) (Haarnoja et al., 2018a), have demonstrated substantial improvements in both performance and sample efficiency in previous on-policy methods.

Neural architecture search Neural architecture search methods aim to automatically search for a good neural architecture for various tasks, such as image classification (Krizhevsky et al., 2012) and segmentation (Liu et al., 2019b), in order to ease the burden of hand-crafted design of dedicated architectures for specific tasks. Several approaches have been proposed to tackle the NAS problem. Zoph and Le (Zoph and Le, 2017) proposed a reinforcement learning-based method that trains an RNN controller to design the neural network (Zoph and Le, 2017). Guo et al. (Guo et al., 2019b) exploited a novel graph convolutional neural networks for policy learning in reinforcement learning. Further successfully introduced approaches include evolutionary algorithm based methods(Real et al., 2017), differentiable methods (Liu et al., 2019c) and one-shot learning methods (Brock et al., 2017; Liu et al., 2019c). Early works of RL-based NAS algorithms (Xie et al., 2018; Pham et al., 2018; Zoph and Le, 2017; Liu et al., 2018) proposed to optimize the entire trajectory (i.e., the entire neural architecture). To the best of our knowledge, most of the previously proposed RL-based NAS algorithms used on-policy RL algorithms, such as REINFORCE or PPO, except (Zhong et al., 2018) which uses Q-learning algorithm for NAS, which is a value-based method and only supports discrete state space problems. For on-policy algorithms, since each update requires new data collected by the current policy and the reward is based on the internal neural network architecture training, the on-policy training of RL-based NAS algorithms inevitably becomes computationally expensive.

GANs architecture search Due to the specificities of GAN and their challenges, such as instability and mode collapse, the NAS approaches proposed for discriminative models cannot be directly transferred to the architecture search of GANs. Only recently, few approaches have been introduced tackling the specific challenges of the GAN architectures. Recently, AutoGAN has introduced a neural architecture search for GANs based on reinforcement learning (RL), thereby enabling a significant speedup of the process of architecture selection (Gong et al., 2019). The AutoGAN algorithm is based on on-policy reinforcement learning. The proposed multi-level architecture search (MLAS) aims at progressively finding well-performing GAN architectures and completes the task in around 2 GPU days. Similarly, AGAN (Wang and Huan, 2019) uses reinforcement learning for generative architecture search in a larger search space. The computational cost for AGAN is comparably very expensive (1200 GPU days). In addition, AdversarialNAS (Gao et al., 2019) and DEGAS (Doveh and Giryes, 2019) adopted a different approach, i.e., differentiable searching strategy (Liu et al., 2019c), for the GAN architecture search problem.

5.3 Preliminary

In this section, we briefly review the basic concepts and notations used in the following sections.

5.3.1 Generative adversarial networks

The training of GANs involves an adversarial competition between two players, a generator and a discriminator. The generator aims at generating realistic-looking images to 'fool' its opponent. Meanwhile, the discriminator aims to distinguish whether an image is real or fake. This can be formulated as a min-max optimization problem:

$$\min_G \max_D \mathbb{E}_{x\sim p_{real}}[\log D(x)] + \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z)))], \tag{5.1}$$

where G and D are the generator and discriminator parametrized by neural networks. z is sampled from random noise. x are the real and $G(z)$ are the generated images.

5.3.2 Reinforcement learning

A Markov decision process (MDP) is a discrete-time stochastic control process. At each time step, the process is in some state s, and its associated decision-maker chooses an available action a. Given the action, the process moves into a new state s' at the next step, and the agent receives a reward.

An MDP could be described as a tuple (S, A, r, P, ρ), where S is the set of states that is able to precisely describe the current situation, A is the set of actions, $r(s, a)$ is the reward function, $P(s'|s, a)$ is the transition probability function, and $\rho(s)$ is the initial state distribution.

MDPs can be particularly useful for solving optimization problems via reinforcement learning. In a general reinforcement learning setup, an agent is trained to interact with the environment and get a reward from its interaction. The goal is to find a policy π that maximizes the cumulative reward $J(\pi)$:

$$J(\pi) = \mathbb{E}_{\tau\sim\rho_\pi} \sum_{t=0}^{\infty} r(s_t, a_t) \tag{5.2}$$

While the standard RL merely maximizes the expected cumulative rewards, the maximum entropy RL framework considers a more general objective (Ziebart, 2010), which favors stochastic policies. This objective shows a strong connection to the exploration-exploitation trade-off, and could also prevent the policy from getting stuck in local optima. Formally, it is given by

$$J(\pi) = \mathbb{E}_{\tau\sim\rho_\pi} \sum_{t=0}^{\infty} [r(s_t, a_t) + \beta\mathcal{H}(\pi(\cdot|s_t))], \tag{5.3}$$

where β is the temperature parameter that controls the stochasticity of the optimal policy.

5.4 Problem formulation

5.4.1 Motivation

Given a fixed search space, GAN architecture search agents aim to discover an optimal network architecture on a given generation task. Existing RL methods update the policy network by using batches of entire architectures sampled from the current policy. Even though these data samples are only used for the current update step, the sampled GAN architectures nevertheless require tedious training and evaluation processes. The sampling efficiency is therefore very low resulting in limited learning progress of the agents. Moreover, the entire architecture sampling leads to a high variance, which might influence the stability of the policy update.

The key motivation of the proposed methodology is to stabilize and accelerate the learning process by step-wise sampling instead of entire-trajectory-based sampling and making efficient use of past experiences from previous policies. To achieve this, we propose to formulate the GAN architecture search problem as an MDP and solve it by off-policy reinforcement learning.

5.4.2 GANs architecture search formulated as MDP

We propose to formulate the GAN architecture search problem as a Markov decision process (MDP) which enables state-based sampling. It further boosts the learning process and overcomes the potential challenge of a large variance stemming from sampling entire architectures that makes it inherently difficult to train a policy using off-policy data.

Formulating GAN architecture search problem as an MDP provides a mathematical description of architecture search processes. An MDP can be described as a tuple (S, A, r, P, ρ), where S is the set of states that is able to precisely describe the current architecture (such as the current cell number, the structure or the performance of the architectures), A is the set of actions that defines the architecture design of the next cell, $r(s, a)$ is the reward function used to define how good the architecture is, $P(s'|s, a)$ is the transition probability function indicating the training process, and $\rho(s)$ is the initial architecture. We define a cell as an architecture block we are using to search in one step. The design details of states, actions, and rewards is discussed in Section 5.5.

It is important to highlight that the formulation proposed in this paper has two main differences compared to previous RL methods for neural architecture search. Firstly, it is essentially different to the classic RL approaches for NAS (Zoph and Le, 2017), which formulate the task as an optimization problem over the entire trajectory/architecture. Instead, the MDP formulation proposed here enables us to do the optimization based on the disentangled steps of cell design. Secondly, it is also different to the progressive formulation used by AutoGAN (Gong et al., 2019), where the optimization is based on the best performance of the current architecture level and can potentially lead to a local minimum solution. Instead, the proposed formulation enables us to potentially conduct a more global optimization using cumulative reward without the burden of calculating the reward over the full trajectory at once. It is important to point out that the multi-level optimization formulation used in AutoGAN (Gong et al., 2019) does not have this property.

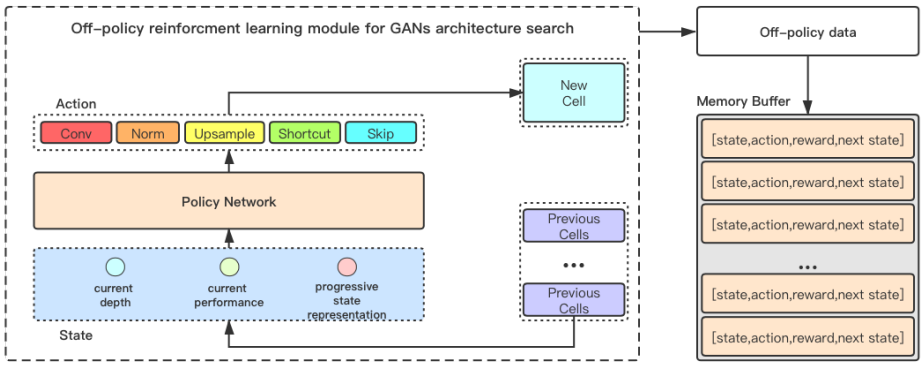

Figure 5.1: Overview of the proposed E^{2}GAN: the off-policy reinforcement learning module for GAN architecture search. The entire process comprises five steps: 1)The agent observes the current state s_t, which is designed as s=[Depth, Performance, Progressive state representation]. 2)The agent makes a decision a_t on how to design the cell added to previous cells according to the state information. a_t includes the skip options, upsampling operations, shortcut options, different types of convolution blocks and a normalization block 3)Progressively train the new architecture, obtain the reward r_t and the new state s_{t+1} information and then loop over it again. 4)Save the off-policy memory tuple $[s_t, a_t, r_t, s_{t+1}]$ into the memory buffer. 5)Sample a batch of data from the memory buffer to update the policy network.

5.5 Off-policy RL for GANs architecture search

In this section, we integrate off-policy RL in the GAN architecture search by making use of the newly proposed MDP formulation. We introduce several innovations to address the challenges of an off-policy learning setup.

The MDP formulation of GAN architecture search enables us to use off-policy reinforcement learning for a step-wise optimization of the entire search process to maximize the cumulative reward.

5.5.1 RL for GANs architecture search

Before we move on to the off-policy solver, we need to design the state, reward, and action to meet the requirements of both the GAN architecture design, as well as of the MDP formulation.

State

MDP requires a state representation that can precisely represent the current network up to the current step. Most importantly, this state needs to be stable during training to avoid adding more variance to the training of the policy network. The stability requirement is particularly relevant since the policy network relies on it to design the next cell. The design of the state is one of the main challenges we face when adopting off-policy RL to GAN architecture search.

Inspired by the progressive GAN (Karras et al., 2017), which has shown to improve generation quality in intermediate RGB outputs of each architecture cell, we propose a progressive state representation for GAN architecture search. Specifically, given a fixed batch of input noise, we adopt the average output of each cell as the progressive state representation. We down-sample this representation to impose a constant size across different cells. Note that there are alternative ways to encode the network information. For example, one could also deploy another network to encode the previous layers. However, we find the proposed design efficient and also effective.

In addition to the progressive state representation, we also use network performance (Inception Score / FID) and layer number to provide more information about the state. To summarize, the designed state s includes the depth, performance of the current architecture, and the progressive state representation.

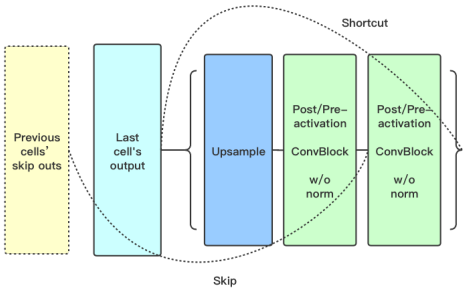

Figure 5.2: The search space of a generator cell in one step. The search space is directly taken from AutoGAN (Gong et al., 2019).

Action

Given the current state, which encodes the information about previous layers, the policy network decides on the next action. The action describes the architecture of one cell. For example, if we follow the search space used by AutoGAN (Gong et al., 2019), action will contain skip options, upsampling operations, shortcut options, different types of convolution blocks, and the normalization option, as shown in Figure 5.2.

This can then be defined as $a = [conv, norm, upsample, shortcut, skip]$. The action output by the agent will be carried out by a softmax classifier decoding into an operation. To demonstrate the effectiveness of our off-policy methods and enable a fair comparison, in all of our experiments, we use the same search space as AutoGAN (Gong et al., 2019), which means we search for generator cells, and the discriminator architecture is pre-designed and growing as the generator becomes deeper. More details on the search space are discussed in Section 5.6.

Reward

We propose to design the reward function as the **performance improvement** after adding the new cell. In this work, we use both Inception Score (IS) and Frchet Inception Distance (FID) as the indicators of the network performance. Since IS score is progressive (the higher the better) and FID score is degressive (the lower the better), the proposed reward function can be formulated as:

$$R_t(s,a) = IS(t) - IS(t-1) + \alpha(FID(t-1) - FID(t)), \tag{5.4}$$

where α is a factor to balance the trade-off between the two indicators. We use $\alpha = 0.01$ in our main experiments. The motivation behind using a combined reward is based on an empirical finding indicating that IS and FID are not always consistent with each other and can lead to a biased choice of architectures. A detailed discussion about the choice of indicators is provided in Section 5.7.

By employing the performance improvement in each step instead of only using performance as proposed in (Gong et al., 2019), RL can maximize the expected sum of rewards over the entire trajectory. This enables us to target the potential global optimal structure with the highest reward:

$$J(\pi) = \sum_{t=0} \mathbb{E}_{(s_t,a_t)\sim p(\pi)} R(s_t,a_t) = \mathbb{E}_{architecture\sim p(\pi)} IS_{final} - \alpha FID_{final}, \tag{5.5}$$

where IS_{final} and FID_{final} are the final scores of the entire architecture.

5.5.2 Off-policy RL solver

The proposed designs of state, reward, and action fulfill the criteria of MDPs and makes it possible to stabilize the training using off-policy samples. We are now free to choose any off-policy RL solver to improve data efficiency.

In this paper, we apply the off-the-shelf soft actor-critic algorithm (SAC) (Haarnoja et al., 2018a), an off-policy actor-critic deep RL algorithm based on the maximum entropy reinforcement learning framework, as the learning algorithm. It has demonstrated to be 10 to 100 times more data-efficient compared to any other on-policy algorithms on traditional RL tasks. In SAC, the actor aims at maximizing expected reward while also maximizing entropy. This increases training stability significantly and improves the exploration during training.

For the learning of the critic, the objective function is defined as:

$$J(Q) = \mathbb{E}_{(s,a)\sim\mathcal{D}} \left[\frac{1}{2} (Q(s,a) - Q_{target}(s,a))^2 \right] \tag{5.6}$$

where Q_{target} is the approximation target for Q :

$$Q_{target}(s,a) = Q(s,a) + \gamma Q_{target}(s', f(\epsilon, s')) \tag{5.7}$$

The objective function of the the policy network is given by:

$$J(\pi) = \mathbb{E}_{\mathcal{D}} \left[\beta[\log(\pi_\theta(f_\theta(\epsilon,s)|s))] - Q(s, f_\theta(\epsilon,s)) \right] \tag{5.8}$$

Input hyperparameters, learning rates α_{ϕ_Q},α_θ
Randomly initialize a Q network $Q(s,a)$ and policy network $\pi(a|s)$ with parameters ϕ_Q, θ and the Lagrange multipliers β,
Initialize the parameters of target networks with $\overline{\phi}_Q \leftarrow \phi_Q$, $\overline{\theta} \leftarrow \theta$
for each iteration **do**
  Reset the weight and cells of E^{2}GAN
  for each time step **do**
    if Exploration **then**
      Sample a_t from $\pi(s)$, add the corresponding cell to E^{2}GAN
    else if Exploitation **then**
      Choose the best a_t from $\pi(s)$ and add the corresponding cell to E^{2}GAN
    end if
    Progressively train the E^{2}GAN
    Observe s_{t+1}, r_t and store (s_t, a_t, r_t, s_{t+1}) in $\mathcal{D}$
  end for
  for each update step **do**
    Sample mini-batches of transitions from $\mathcal{D}$ and update Q and π with gradients
    Update the target networks with soft replacement:

$$\overline{\phi}_Q \leftarrow \tau\phi_Q + (1-\tau)\overline{\phi}_Q$$
$$\overline{\theta} \leftarrow \tau\theta + (1-\tau)\overline{\theta}$$

  end for
end for

Algorithm 3: Pseudo code for E^{2}GAN search

where π_θ is parameterized by a neural network f_θ, ϵ is an input vector consisting of Gaussian noise, and the $\mathcal{D} \doteq \{(s, a, s', r)\}$ is the replay buffer for storing the MDP tuples (Mnih et al., 2015). β is a positive Lagrange multiplier that controls the relative importance of the policy entropy versus the safety constraint.

5.5.3 Implementation of E^{2}GAN

In this section, we present the implementation details of the proposed off-policy RL framework E^{2}GAN. The training process is briefly outlined in Algorithm 3.

Agent training

Since we reformulated the NAS as a multi-step MDP, our agent will make several decisions in any trajectory $\tau = [(s_1, a_1), ...(s_n, a_n)]$. In each step, the agent will collect this experience $[s_t, a_t, r_t, s_{t+1}]$ in the memory buffer $\mathcal{D}$. Once the threshold of the smallest memory length is reached, the agent is updated using the Adam (Kingma and Ba, 2014) optimizer via the objective function presented in Eq. 5.8 by sampling a batch of data from the memory buffer $\mathcal{D}$ in an off-policy way.

The entire search comprises two periods: the exploration period and the exploitation period. During the exploration period, the agent will sample any possible architecture. While in the exploitation period, the agent will choose the best architecture, in order to quickly stabilize the policy.

The exploration period lasts for 70% of iterations, and the exploitation takes 30% iterations. Once the memory threshold is reached, for every exploration step, the policy will be updated once. For every exploitation step, the policy will be updated 10 times in order to converge quickly.

Proxy task

We use a progressive proxy task in order to collect the rewards fast. When a new cell is added, we train the current full trajectory for one epoch and calculate the reward for the current cell. Within a trajectory, the previous cells' weights will be kept and trained together with the new cell. In order to accurately estimate the Q-value of each state-action pair, we reset the weight of the neural network after finishing the entire architecture trajectory design.

5.6 Experiments

5.6.1 Dataset

In this paper, we use the CIFAR-10 dataset (Krizhevsky, Hinton, et al., 2009) to evaluate the effectiveness and efficiency of the proposed E^{2}GAN framework. The CIFAR-10 dataset consists of 50,000 training images and 10,000 test images with a 32×32 resolution. We use its training set without any data augmentation technique to search for the architecture with the highest cumulative return for a GAN generator. Furthermore, to evaluate the transferability of the discovered architecture, we also adopt the STL-10 dataset (Coates et al., 2011) to train the network without any other data augmentation to make a fair comparison to previous works.

5.6.2 Search space

To verify the effectiveness of the off-policy framework and to enable a fair comparison, we use the same search space as used in the AutoGAN experiments (Gong et al., 2019). There are five control variables: 1)Skip operation, which is a binary value indicating whether the current cell takes a skip connection from any specific cell as its input. Note that each cell could take multiple skip connections from other preceding cells. 2)Pre-activation (He et al., 2016) and post-activation convolution block. 3)Three types of normalization operations, including batch normalization (Ioffe and Szegedy, 2015), instance normalization (Ulyanov et al., 2016), and no normalization. 4)Upsampling operation which is standard in current image generation GAN, including bi-linear upsampling, nearest neighbor upsampling, and stride-2 deconvolution. 5)Shortcut operation.

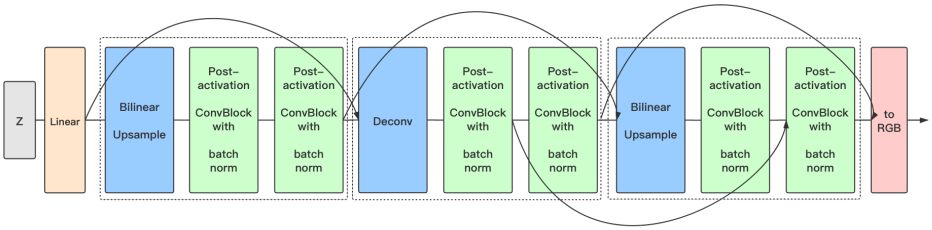

Figure 5.3: The generator architecture discovered by E^{2}GAN on CIFAR-10.

5.6.3 Results

The generator architecture discovered by E^{2}GAN on the CIFAR-10 training set is displayed in Figure 5.3. For the task of unconditional CIFAR-10 image generation (no class labels used), several notable observations could be summarized:

* E^{2}GAN prefers post-activation convolution block to pre-activation convolution blocks. This finding is contrary to AutoGAN's preference, but coincides with previous experiences from human experts.

* E^{2}GAN prefers the use of batch normalization. This finding is also contrary to AutoGAN's choice, but is in line with experts' common practice.

* E^{2}GAN prefers bi-linear upsample to nearest neighbour upsample. This in theory provides finer upsample ability between different cells.

Our E^{2}GAN framework only takes about 0.3 GPU day for searching while the AGAN spends 1200 GPU days and AutoGAN spends 2 GPU days.

We train the discovered E^{2}GAN from scratch for 500 epochs and summarize the IS and FID scores in Table 5.1. On the CIFAR-10 dataset, our model achieves a **highly competitive FID 11.26** compared to published results by AutoGAN (Gong et al., 2019), and hand-crafted

Methods	Inception Score	FID	Search Cost (GPU days)
DCGAN (Radford et al., 2015)	$6.64 \pm .14$	-	*
Improved GAN (Salimans et al., 2016)	$6.86 \pm .06$	-	*
LRGAN (Yang et al., 2017)	$7.17 \pm .17$	-	*
DFM (Warde-Farley and Bengio, 2016)	$7.72 \pm .13$	-	*
ProbGAN (He et al., 2019)	7.75	24.60	*
WGAN-GP, ResNet (Gulrajani et al., 2017)	$7.86 \pm .07$	-	*
Splitting GAN (Grinblat et al., 2017)	$7.90 \pm .09$	-	*
SN-GAN (Miyato et al., 2018)	$8.22 \pm .05$	$21.7 \pm .01$	*
MGAN (Hoang et al., 2018)	$8.33 \pm .10$	26.7	*
Dist-GAN (Tran et al., 2018)	-	$17.61 \pm .30$	*
Progressive GAN (Karras et al., 2017)	**8.80 ± .05**	-	*
Improv MMD GAN (Wang et al., 2019b)	8.29	16.21	*
Random search-1 (Gong et al., 2019)	8.09	17.34	-
Random search-2 (Gong et al., 2019)	7.97	21.39	-
AGAN (Wang and Huan, 2019)	$8.29 \pm .09$	30.5	1200
AutoGAN-top1 (Gong et al., 2019)	$8.55 \pm .10$	12.42	2
AutoGAN-top2 (Gong et al., 2019)	$8.42 \pm .07$	13.67	2
AutoGAN-top3 (Gong et al., 2019)	$8.41 \pm .11$	13.68	2
E^{2}GAN-top1	$8.51 \pm .13$	**11.26**	0.3
E^{2}GAN-top2	$8.50 \pm .09$	12.96	0.3
E^{2}GAN-top3	$8.42 \pm .11$	12.48	0.3

Table 5.1: Inception score and FID score of unconditional image generation task on CIFAR-10. We achieve **a highly competitive FID of 11.26** compared to published works. We mainly compare our approach with RL-based NAS approaches: AGAN (Wang and Huan, 2019) and AutoGAN (Gong et al., 2019). Architectures marked by (*) are manually designed.

Methods	Inception Score	FID	Search Cost (GPU days)
D2GAN (Nguyen et al., 2017)	7.98	-	Manual
DFM (Warde-Farley and Bengio, 2016)	$8.51 \pm .13$	-	Manual
ProbGAN (He et al., 2019)	$8.87 \pm .095$	46.74	Manual
SN-GAN (Miyato et al., 2018)	$9.10 \pm .04$	$40.1 \pm .50$	Manual
Dist-GAN (Tran et al., 2018)	-	36.19	Manual
Improving MMD GAN (Wang et al., 2019b)	9.34	37.63	Manual
AGAN (Wang and Huan, 2019)	$9.23 \pm .08$	52.7	1200
AutoGAN-top1 (Gong et al., 2019)	$9.16 \pm .13$	31.01	2
E^{2}GAN-top1	**9.51 ± .09**	**25.35**	0.3

Table 5.2: Inception score and FID score for the unconditional image generation task on STL-10. E^{2}GAN uses the discovered architecture on CIFAR-10. Performance is significantly better than other RL-based competitors.

Figure 5.4: The generated CIFAR-10(left) and STL-10(right) results of E^{2}GAN which are randomly sampled without cherry-picking.

GAN (Radford et al., 2015; Salimans et al., 2016; Yang et al., 2017; Warde-Farley and Bengio, 2016; He et al., 2019; Gulrajani et al., 2017; Grinblat et al., 2017; Miyato et al., 2018; Hoang et al., 2018; Wang et al., 2019b). In terms of IS score, E^{2}GAN is also highly competitive to AutoGAN (Gong et al., 2019). We additionally report the performance of the top2 and top3 architectures discovered in one search. Both have higher performance than the respective AutoGAN counterparts.

We also test the transferability of E^{2}GAN. We retrain the weights of the discovered E^{2}GAN architecture using the STL-10 training and unlabeled set for the unconditional image generation task. **E^{2}GAN achieves a highly-competitive performance on both IS (9.51) and FID (25.35)**, as shown in Table 5.2.

Because our main contribution is the new formulation and using off-policy RL for GAN architecture framework, we compare the proposed method directly with existing RL-based algorithms. We use the exact same searching space as AutoGAN, which does not include the search for a discriminator. As GAN training is an interactive procedure between generator and discriminator, one might expect better performance if the search is conducted on both networks. We report our scores using the exact same evaluation procedure provided by the authors of AutoGAN. The reported scores are based on the best models achieved during training on a 20 epoch evaluation interval. Mean and standard deviation of the IS score are calculated based on the 10-fold evaluation on 50,000 generated images. We additionally report the performance curve against training steps of E^{2}GAN and AutoGAN for three runs in the supplementary material.

5.7 Discussion

5.7.1 Reward choice: IS and FID

IS and FID scores are two main evaluation metrics for GAN. We conduct the ablation study of using different combinations. Specifically, IS only ($\alpha = 0$) and the combination of IS and FID ($\alpha = 0.01$) as the reward. Our agent successfully discovered two different architectures. When only IS is used as the reward, the agent discovered a different architecture using only

Methods	Inception Score	FID	Search Cost (GPU days)
AutoGAN-top1 (Gong et al., 2019)	$8.55 \pm .1$	12.42	2
E^{2}GAN(IS and FID as reward)	$8.51 \pm .13$	11.26	0.3
E^{2}GAN(IS only as reward)	$\mathbf{8.81 \pm .11}$	15.64	0.1

Table 5.3: Performance on the unconditional image generation task for CIFAR-10 with different reward choices.

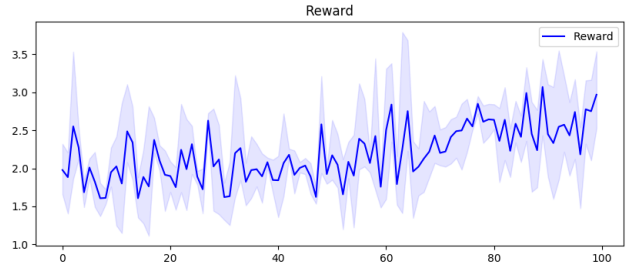

Figure 5.5: Training curves on architecture searching. IS score on the proxy task against training time steps. E^2GAN shows relatively good stability and reproducibility.

0.1 GPU day. The searched architecture achieved state-of-the-art IS score of 8.86, as shown in Table 5.3., but a relatively plain FID of 15.78. This demonstrates the effectiveness of the proposed method, as we are encouraging the agent to find the architecture with a higher IS score. Interestingly, this shows that, at least in certain edge cases, the IS and FID may not always have a strong positive correlation. This finding motivates us to include FID in the reward. When both IS and FID are used as the reward signal, the discovered architecture performs well in term of both metrics. This combined reward takes 0.3 GPU days (compared to 0.1 GPU days of IS only optimization) because of the relatively expensive cost of FID computation.

5.7.2 Reproducibility

We train our agent over 3 different seeds. As shown in Figure 5.5, we observe that our agent steadily converged the policy in the exploitation period. E^2GAN can find similar architectures with relatively good performance on the proxy task.

5.8 Conclusion

We proposed a novel off-policy RL method, E^2GAN, to efficiently and effectively search for GAN architectures. We reformulated the problem as an MDP process, and overcame the challenges of using off-policy data. We first introduced a new progressive state representation. We additionally introduced a new reward, which allowed us to target the potential global optimization in our MDP formulation. The E^2GAN achieves state-of-the-art efficiency in GAN architecture searching, and the discovered architecture shows highly competitive performance.

Acknowledgement

The contributions of Yuan Tian, Qin Wang, and Olga Fink were funded by the Swiss National Science Foundation (SNSF) Grant no. PP00P2_176878.

6 Discussions

In this dissertation, an immediate-level prescriptive maintenance framework is proposed with the aim of improving the system performance and prolonging the RUL. Detailed discussions for the individual studies have been presented in the respective chapters. In Chapter 2, we propose a real-time model calibration module, Chapter 3 proposes an end-to-end load allocation module for multi-battery systems, Chapter 3 introduces a novel opponent modeling method with an effective MARL for the maintenance coordination problems, and in Chapter 5, we develop an efficient black-box optimization module that can be applied to fast process optimization and system reconfiguration tasks. This chapter reviews the main elements of the thesis and discusses the key findings of this research.

6.1 Real-time model calibration with reinforcement learning

Model calibration approaches aim to close the gap between the performance model and the measured data. However, previous approaches are computationally expensive and cannot be applied in real-time or require a large amount of labeled data. In Chapter 2, a real-time degradation calibration module is proposed to compensate for the deviation between the physics-based performance model and the data measured from the degraded real system. This module plays an essential role in the proposed framework. It cannot only calibrate the performance model, but most importantly, it provides an indispensable understanding of the degradation state for the subsequent steps of controlling the degradation process. The discussion of the key findings is summarized below.

Inference accuracy In Chapter 2, we compare the inference performance of the proposed RL-based method to a model-based approach, the UKF, and a data-based supervised learning approach on two open prognostics datasets: N-CMAPSS and AGTF30. The proposed approach showed the best inference accuracy on both datasets, achieving an RMSE of $3.30e-04$ RMSE on AGTF30 and $2.50e-03$ on N-CMAPSS. It is worth mentioning that unlike supervised learning, which needs the ground truth degradation parameters for training, our method does not need any prior knowledge about the degradation parameters.

Computational cost Another interesting finding is related to the computational cost. The proposed method, which is a data-driven approach without any online optimization, provides better inference accuracy with a speed-up of $\times 150$ compared to the model-based UKF. The proposed method shows great potential for applications that require real-time inference.

Robustness One of the essential requirements for parameter inference in real-world scenarios is the robustness to the sensor noise and environmental uncertainty. In Chapter 2, the proposed method can provide a very good inference, achieving an RMSE of $3.30e-04$ under model bias and an RMSE of $4.22e-04$ under model noise. This demonstrates better robustness compared to UKF, which achieves an RMSE of $2.04e-3$ under model noise and failed to optimize a stable inference under model noise. We demonstrate that the proposed approach is more reliable and better applicable to real-world scenarios.

Reinforcement learning as probabilistic inference To leverage the application of RL for model calibration or parameter inference tasks, we formulated the model calibration task as an inverse problem of a tracking problem. Under this formulation, RL methods become

an alternative solution for inference problems. Surprisingly, this is in line with the idea from a previous theoretical research (Levine, 2018), which demonstrated that the Markov decision process could be formulated as a probabilistic inference problem. To the best of our knowledge, this is the first time that the application of RL as a probabilistic inference has been demonstrated on real-world applications.

Action stabilization is necessary for inference tasks Another important finding is that the action stabilization term in the policy update objective can improve the performance and robustness of the proposed method. Since the agent is trained without labeled data, the agent can only minimize the tracking loss instead of maximizing the inference accuracy. Without any constraint, the action may exhibit a large variance. Such a situation is undesirable in many real-world applications where it is important to obtain a stable and smooth action over time. Therefore, in order to stabilize the actions, we introduce the CLAC algorithm, a modification of the LAC algorithm, which significantly improves the action stability under model uncertainty and sensor noise. The proposed stabilization term can be applied to other tasks which require stability and smoothness of actions, for example, robotics movement control (Taylor et al., 2021) and trajectory tracking tasks (Han et al., 2020).

6.2 End-to-end load allocation in real-time with reinforcement learning

In Chapter 3, we proposed an end-to-end load allocation framework that is able to infer the health state of the batteries implicitly from raw signal measurements. Based on this implicit inference, the algorithm is able to allocate the load to each of the batteries in real-time with the aim to prolong the working cycle of the deployed multi-battery systems. The proposed framework shows considerable scalability on NASA multi-battery system models (Daigle and Kulkarni, 2013). Moreover, we proposed the novel Dirichlet policy to tackle the simplex constraint in allocation tasks. This module provides a feasible solution but also a promising direction for prolonging RUL by prescriptive operation. The discussion of the key findings is summarized below.

End-to-end prescriptive operation with reinforcement learning In Chapter 3, we proposed an RL-based module that can perform end-to-end load allocation for multi-battery systems. The developed framework only requires raw current and voltage measurements, along with the incoming power demand, as inputs. Moreover, the information is processed automatically via the trained neural network, and the module can handle data streams in real-time. To the best of our knowledge, this is the first time an algorithm has been shown to be capable of directly performing the load allocation strategy in an end-to-end way. We evaluated the performance of the proposed module on different multi-battery systems that were modeled by a simulator (Daigle and Kulkarni, 2013). According to the experimental results, the proposed module was able to effectively prolong the working cycle of a four-battery system by 15.2% on average compared to the equal distribution allocation policy. Also, the proposed module showed considerable transferability to a new system configuration. Following the same setup as for the system with four batteries, the module can be trained and deployed on an eight-battery system, prolonging the working cycle by 31.9% on average. When deployed on second-life battery systems, the learned strategy can prolong the working cycle by 151.0 % on average. This module has demonstrated that RL is able to infer the health state implicitly without any explicit information on the degradation condition and is able to integrate this into control. This is a very promising direction for many different applications to prolong their working cycle and RUL.

Dirichlet policy for allocation tasks In Chapter 3, we proposed the Dirichlet policy as a plug-in module for RL algorithms to tackle continuous action space allocation tasks.

We find that the proposed method is bias-free and has lower variance in the policy gradient compared to the Gaussian-softmax policy. The numerical experiments showed consistent results, where the proposed method converged faster and is also more robust to learning rate changes. This result can be applied to a wide range of real-world allocation tasks, for example, the reliability redundancy allocation, which can help improve system reliability and minimize the cost (Wang et al., 2020; Sabri-Laghaie and Karimi-Nasab, 2019).

6.3 Multi-agent coordination in mixed cooperative-competitive environments

In Chapter 4, we proposed a novel objective for opponent modeling with an effective MARL method for mixed-objective MAS tasks. This module can be applied as a complementary module when the structural and operational dependencies within machines and systems are required to be considered. The discussion of the key findings is summarized below.

Opponent model does matter In this research, we find that by deriving a lower bound of the log objective of an individual agent, accurate opponent predictions can alleviate the non-stationarity problem of policy updates in MARL. Moreover, with such accurate opponent predictions, each agent can access a more reliable value estimation. This provides then a better guidance for its own policy update. In addition, it allows the agent to become a better collaborator or stronger adversary to influence other agents in cooperative and competitive settings, respectively. As a consequence, all the policies can improve iteratively. We validated this hypothesis on a challenging MAS benchmark with mixed-objective and continuous state and action space. The proposed opponent model has shown a superior prediction accuracy compared to two strong baselines, the ROMMEO (Tian et al., 2019)and the PR2 (Wen et al., 2019). With better opponent predictions, the proposed MARL algorithm achieved equivalent performance on cooperative tasks and state-of-the-art performance on mixed-objective tasks compared to the baselines. This result supports the conclusion from previous studies that the opponent model does matter (Wen et al., 2018).

Extension of the proposed MARL algorithm to maintenance applications One of the potential application fields for the proposed methodology is the maintenance scheduling or operation of power grids, where the generation units operate in a competitive environment. In such problems, the decision of each generation unit will inevitably influence the entire network. For example, the maintenance scheduling in electricity markets, where the generators are required to find an optimal bidding strategy and maintenance schedule to maximize profit and improve the reliability of the entire system. However, there is a trade-off between the reliability and the profit of the system since the maintenance imposes cost and takes time during which the generation units cannot produce energy (Rokhforoz and Fink, 2021; Rokhforoz et al., 2021). Thus, the strategy of each unit should take the potential operations of other units into account. Such problems can be modeled by MAS. As discussed before, centralized methods suffer from high computational costs and are less flexible, the proposed decentralized MARL coordination module can be a promising alternative solution for such problems.

6.4 Effective and efficient black-box optimization via reinforcement learning

In Chapter 5, we propose an effective and efficient method for optimization tasks, especially for derivative-free problems, where the structure of the objective function and the constraints are not known and cannot be unexplored or may not even exist. This work is the final piece in the immediate-level prescriptive maintenance framework proposed in this thesis. The difference between the previous three modules and this contribution is that this module provides a solution for decisions that are only made once and results in a unique solution, such as system reconfiguration and process optimization. The previous three modules focused

on sequential decision-making tasks where the decisions are made sequentially. The discussion of the key findings is summarized below.

MDP formulation for BBO The most important finding of this module is that the MDP formulation can significantly improve the searching efficiency. We evaluated the proposed module on a challenging black-box optimization task, the GAN architecture search on the CIFAR-10 and STL-10 datasets. The proposed framework can provide reliable recommendations with only a limited number of samples (300 samples in this example). Compared to the SOTA algorithm, the AutoGAN (Gong et al., 2019), the proposed algorithm is seven times faster in successfully finding a better neural architecture, becoming a new SOTA. The found architecture also becomes the new SOTA architecture for CIFAR-10 and STL-10 datasets.

Extension to maintenance applications In Chapter 5, with the superior performance on GAN architecture search, the proposed idea has provided a new direction for RL-based optimization. Moreover, it has great potential for a wide range of real-world BBO tasks in maintenance applications. For example, the reconfiguration of power grid after physical failures and severe power disruptions is a classical BBO task in reliability engineering, where the function between the network topology and the performance is not known. Such a problem aims to find the optimal network topology to improve the remaining network's largest amount of power supply (LPS), which is also a unique decision in any unique scenario of a severe power disruption (Zhang et al., 2021d). And a quick power recovery in extreme events can have a significant impact on society. However, the conventional nonlinear interior point method is only feasible for a fixed topology (Zhang et al., 2021d). We provide a promising alternative solution, which does not depend on the objective function but solves the problem by maximizing the reward. Moreover, other optimization tasks, such as the optimization of the power grid, and supply chain networks with the purpose of improving the system resilience and performance, are also potentially promising applications for the proposed methodology. Also, the GAN architecture search module can be directly applied to search for the optimal GAN architecture with the aim of data generation for fault diagnostics or prognostics.

6.5 Proposed prescriptive maintenance and operation framework

With the proposed modules, the prescriptive maintenance and operation framework can be applied to complex real-world tasks. For example, in the UAVs fleet coordination task, each UAV and its batteries may degrade differently. The different degradation states can be calibrated by the proposed model calibration module. For each UAV, the maintenance should take the overall availability of the entire fleet and the maintenance cost into account, which can be obtained by the multi-agent coordination module. When performing a flight, the load allocation module can prolong the working cycle and thereby improve reliability and deployability. Lastly, for each flight, the trajectory can be optimized by the fast optimization module with the aim of the lowest consumption. The proposed framework is general and flexible, with considerable scalability and transferability, showing great potential for solving a wide range of maintenance tasks.

7 Conclusions

This chapter revisits the main aim and objectives of this dissertation. Key conclusions are then discussed to answer the main research questions. Finally, the limitations and future research directions are discussed.

7.1 Research objectives revisited

The aim of this research was to *develop an immediate-level prescriptive maintenance framework that integrates degradation awareness into decision-making to improve the system performance, reliability, and availability, and further prolong the working cycle and the RUL*

This aim was accomplished by developing the framework that comprises four modules: (1) The development of a real-time degradation calibration module that is able to calibrate the degradation parameters of physics-based performance models in real-time (Chapter 2); (2) The development of an immediate-level end-to-end load allocation module with a novel Dirichlet policy with deep RL that can prolong the working cycle and RUL of multi-battery systems (Chapter 3); (3) The development of a distributed multi-agent coordination module that can effectively obtain feasible policies under cooperative, competitive and mixed cooperative-competitive environments via a novel time dynamical opponent model (Chapter 4); (4) The development of a fast black-box optimization module that can effectively and efficiently provide reliable and reproducible recommendations with limited data (Chapter 5).

7.2 Summary

In this dissertation, the main research question:*How can we integrate degradation into decision-making, and improve the system performance and prolong RUL via immediate-level operations?* has been answered by proposing an immediate-level prescriptive maintenance framework comprising four modules. Moreover, this dissertation pushed the boundary of state-of-the-art research in the fields of prescriptive maintenance and reinforcement learning by answering the following five research questions:

1. How can we capture the implicit degradation state in real-time without supervision? In Chapter 2, we reformulated the model calibration into a tracking problem, which can be seen as the inverse problem of the calibration problem. This formulation unleashed the potential of reinforcement learning methods for inference tasks. We further introduced a novel CLAC algorithm that shows a competitive accuracy on two challenging open benchmarks, the AGTF30 and N-CMAPSS. We compared the proposed DRL approach to UKF and the supervised learning method with respect to several performance evaluation metrics, such as accuracy, robustness, and computational cost. The proposed DRL approach has shown a better inference accuracy than UKF and supervised learning. Compared to UKF, RL methods do not need any online optimization and are significantly faster during inference, which is particularly important for real-time industrial applications, such as the application in operational digital twins. Moreover, the proposed DRL approach has shown to be more robust to sensor noise and model bias. Compared to supervised learning methods, DRL methods neither require labeled data nor any supervision.

2. How can we perform prescriptive load allocation based on only raw sensor measurements in real-time? In Chapter 3, we developed an immediate-level end-to-end load allocation module. For the first time, we successfully prolonged the working cycle

of the deployed multi-battery systems based on only voltage and current information. We demonstrated that, compared to the equally distributed load allocation, the average length of the discharge cycle of the deployed four-battery system can be prolonged by an average of 15.2%. The proposed framework showed considerable scalability that can be easily applied on an eight-battery system without any modification, while prolonging the working cycle by an average of 31.9%. Besides, it is particularly applicable to second-life batteries, and can prolong the working cycle by an average of 151.0%.

3. How can reinforcement learning effectively tackle continuous action space allocation tasks? In Chapter 3, we answered the question above by proposing the Dirichlet policy as an alternative to Gaussian and Gaussian-softmax policy for continuous action space allocation tasks. We found that Gaussian-softmax policy is injective for simplex output, and theoretically and experimentally demonstrated the superior performance of the proposed method. We showed that the Dirichlet policy converges significantly faster resulting in better performance and is also more robust to changes in hyperparameters compared to the Gaussian-softmax policy. The proposed method has been combined with the second module for load allocation that can significantly prolong the working cycle of deployed systems.

4.How can we address the non-stationarity of MARL under mixed-objective scenarios? To answer this question, Chapter 4 demonstrated that by modeling the opponent model parameters as a dynamical system and updating them according to the proposed temporal improvement assumption, we implicitly minimized the KL-divergence between the opponent model prediction and the underlying ground truth. Thus, the proposed method can alleviate the non-stationarity problem in opponent modeling. Most importantly, the proposed opponent model is general and supports cooperative, competitive, and mixed cooperative-competitive environments. Based on the proposed opponent modeling, we further introduced an effective MARL framework and validated the effectiveness on the multi-agent particle environments and the classic differential game.

5. How can reinforcement learning agents learn and give recommendations with limited data in large action- and state-space BBO tasks? To answer this question, in Chapter 5, we proposed to model large action space optimization tasks as MDPs with an adaptive learning strategy. Compared to the conventional bandit formulation, the proposed formulation can achieve a smoother sampling of neural architectures, which enables a more effective RL-based search algorithm. Moreover, compared to the progressive search, we speculate that our formulation can potentially target a global optimum. We investigated this hypothesis on an open GAN architecture search benchmark. In progressive search methods, such as AutoGAN, the GAN architecture of an earlier layer is decided greedily without considering future layers. However, a better neural architecture may have a lower reward for the first layer but a higher final reward. The experimental results are consistent with our hypothesis. In addition, with the tuned exploration and update strategy, we show considerable reproducibility by providing reliable decisions with limited data. By combining the adaptive learning strategy with the proposed MDP formulation, we demonstrated the superior performance on a challenging BBO task, the GAN architecture search.

7.3 Limitations and outlook

This dissertation has explored the immediate-level prescriptive maintenance operations that can improve the performance and reliability of the deployed systems, and can prolong their RUL. The proposed framework integrates degradation awareness into decision-making and provides four modules for typical immediate-level prescriptive maintenance scenarios.

While several contributions both on the methodological as well on the application side have been made in this research, there are still some challenges that we left for future research. This section points out some of these challenges and discusses possible future directions.

Interpretability In Chapter 3, we developed a real-time load allocation framework that allows the agent to infer the system health information implicitly and prescribe allocation actions accordingly. However, since we did not model the system health information explicitly, the lack of interpretability may be an obstacle for some industrial applications. Recently, combining physics-based information with data-driven approaches has shown promising results on prognostic and diagnostic tasks, leading to improvements in performance and training efficiency, as well as to better interpretability. However, this line of research has not been sufficiently explored in the context of reinforcement learning. The development of decision-making approaches that integrate physics-based information could potentially improve interpretability and also support the applicability to more complex systems.

Domain shift For data-driven approaches, it is well known that when a trained model is deployed on unseen operating conditions, the performance can deteriorate significantly (Wang et al., 2019a). This is due to the data distribution difference between training and testing environments. This difference is also referred to as domain shift (Wang et al., 2019a). Domain adaptation(DA) methods have been proposed to tackle this kind of challenges. Domain adaptation methods aim to leverage a small amount of data under new operating conditions and improve the trained model's generalization ability. In this thesis, we did not consider the domain shift between the training environment and the deployed system, which may limit the applicability in real scenarios. Although DA has been broadly applied in computer vision and natural language processing, it has not been extensively studied in the field of reinforcement learning and sequential decision-making. Some prior works of RL with DA only consider the domain shift on the state space, which is insufficient to tackle the operating conditions or degradation state difference. In such cases, the system dynamics changes. To address the dynamics shift problem, it is important to develop corresponding methods that can effectively adapt the policy to the target domain.

Offline RL In this thesis, we mainly adopted off-policy RL to develop the proposed modules. However, the fact that RL provides a fundamentally online learning paradigm is also one of the biggest limitations for industrial applications due to the lack of suitable simulation environments. The conventional RL agent requires to collect experience iteratively by interacting with the environment to learn and update, which is not practical when simulation environments are not available. However, interaction with real industrial systems is expensive and not feasible for safety-critical applications. Recently, offline RL, a purely data-driven approach without the need of interaction with environments, has drawn substantial attention in different fields. Offline RL provides a promising alternative for prescriptive maintenance tasks. This could enable a broader application capability.

Prescriptive maintenance benchmarks In this dissertation, we found that there is an urgent need for open access benchmarks for prescriptive maintenance. Such benchmarks would enable better comparison between the different methods and foster methodological research in that area. Thus, a general prescriptive maintenance environment that covers typical scenarios and tasks in the field would benefit the PHM field and boost the development of prescriptive technologies to a great extent.

Bibliography

Abrate, Carlo, Alessio Angius, Gianmarco De Francisci Morales, Stefano Cozzini, Francesca Iadanza, Laura Li Puma, Simone Pavanelli, Alan Perotti, Stefano Pignataro, and Silvia Ronchiadin (2021). "Continuous-Action Reinforcement Learning for Portfolio Allocation of a Life Insurance Company". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, pp. 237–252.

Albrecht, Stefano V and Peter Stone (2018). "Autonomous agents modelling other agents: A comprehensive survey and open problems". In: *Artificial Intelligence* 258, pp. 66–95.

Amari, Shun-Ichi (1998). "Natural gradient works efficiently in learning". In: *Neural computation* 10.2, pp. 251–276.

Ansari, Fazel, Robert Glawar, and Tanja Nemeth (2019). "PriMa: a prescriptive maintenance model for cyber-physical production systems". In: *International Journal of Computer Integrated Manufacturing* 32.4-5, pp. 482–503.

Ansari, Fazel, Robert Glawar, and Wilfried Sihn (2020). "Prescriptive maintenance of CPPS by integrating multimodal data with dynamic bayesian networks". In: *Machine Learning for Cyber Physical Systems, Technologies for Intelligent Automation* 11, pp. 1–8.

Arel, Itamar, Cong Liu, Tom Urbanik, and Airton G Kohls (2010). "Reinforcement learning-based multi-agent system for network traffic signal control". In: *IET Intelligent Transport Systems* 4.2, pp. 128–135.

Arias Chao, Manuel (2021). "Combining Deep Learning and Physics-Based Performance Models for Diagnostics and Prognostics". PhD thesis. ETH Zurich.

Arias Chao, Manuel, Darrel S. Lilley, Peter Mathé, and Volker Schloßhauer (2015). "Calibration and Uncertainty Quantification of Gas Turbine Performance Models". In: *Proceedings of the ASME Turbo Expo*. Vol. 7A, V07AT29A001. ISBN: 9780791856765. DOI: 10.1115/gt2015-42392.

Bai, Yunfei, Hongwen He, Jianwei Li, Shuangqi Li, Ya-xiong Wang, and Qingqing Yang (2019). "Battery anti-aging control for a plug-in hybrid electric vehicle with a hierarchical optimization energy management strategy". In: *Journal of Cleaner Production* 237, p. 117841.

Bao, Jianmin, Dong Chen, Fang Wen, Houqiang Li, and Gang Hua (2017). "CVAE-GAN: fine-grained image generation through asymmetric training". In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2745–2754.

Barto, Andrew G and Sridhar Mahadevan (2003). "Recent advances in hierarchical reinforcement learning". In: *Discrete event dynamic systems* 13.1-2, pp. 41–77.

Bellman, Richard (1956). "Dynamic programming and Lagrange multipliers". In: *Proceedings of the National Academy of Sciences of the United States of America* 42.10, p. 767.

Bhatnagar, Shalabh, Doina Precup, David Silver, Richard S Sutton, Hamid R Maei, and Csaba Szepesvári (2009). "Convergent temporal-difference learning with arbitrary smooth function approximation". In: *Advances in Neural Information Processing Systems*, pp. 1204–1212.

Björsell, Niclas and Amirhossein Hosseinzadeh Dadash (2021). "Finite Horizon Degradation Control of Complex Interconnected Systems". In: *IFAC-PapersOnLine* 54.1, pp. 319–324.

Borguet, S.J (2012). "Variations on the Kalman Filter for Enhanced Performance Monitoring of Gas Turbine Engines". PhD Thesis. Université de Liège.

Brock, Andrew, Jeff Donahue, and Karen Simonyan (2018). "Large scale gan training for high fidelity natural image synthesis". In: *arXiv preprint arXiv:1809.11096*.

Brock, Andrew, Theodore Lim, James M Ritchie, and Nick Weston (2016). "Neural photo editing with introspective adversarial networks". In: *arXiv preprint arXiv:1609.07093*.

Brock, Andrew, Theodore Lim, James M Ritchie, and Nick Weston (2017). "Smash: one-shot model architecture search through hypernetworks". In: *arXiv preprint arXiv:1708.05344*.

Brown, George W (1951). "Iterative solution of games by fictitious play". In: *Activity analysis of production and allocation* 13.1, pp. 374–376.

Brown, Noam and Tuomas Sandholm (2019). "Superhuman AI for multiplayer poker". In: *Science* 365.6456, pp. 885–890.

Buşoniu, Lucian, Tim de Bruin, Domagoj Tolić, Jens Kober, and Ivana Palunko (2018). "Reinforcement learning for control: Performance, stability, and deep approximators". In: *Annual Reviews in Control*.

Cao, Chengxuan and Ziyan Feng (2020). "Optimal capacity allocation under random passenger demands in the high-speed rail network". In: *Engineering Applications of Artificial Intelligence* 88, p. 103363.

Chao, Manuel Arias, Chetan Kulkarni, Kai Goebel, and Olga Fink (2022). "Fusing physics-based and deep learning models for prognostics". In: *Reliability Engineering & System Safety* 217, p. 107961.

Chao, Manuel Arias, Yuan Tian, Chetan Kulkarni, Kai Goebel, and Olga Fink (2020). "Real-Time Model Calibration with Deep Reinforcement Learning". In: *arXiv preprint arXiv:2006.04001*.

Chen, Hao, Jian Chen, Huaxin Lu, Chizhou Yan, and Zhiyang Liu (2020a). "A modified MPC-based optimal strategy of power management for fuel cell hybrid vehicles". In: *IEEE/ASME Transactions on Mechatronics* 25.4, pp. 2009–2018.

Chen, Yifan, Zhiyong Li, Bo Yang, Ke Nai, and Keqin Li (2020b). "A Stackelberg game approach to multiple resources allocation and pricing in mobile edge computing". In: *Future Generation Computer Systems* 108, pp. 273–287.

Cho, Anthony D, Rodrigo A Carrasco, and Gonzalo A Ruz (2022). "Improving prescriptive maintenance by incorporating post-prognostic information through chance constraints". In: *IEEE Access*.

Choi, Yunjey, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo (2018). "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8789–8797.

Chou, Po-Wei, Daniel Maturana, and Sebastian Scherer (2017). "Improving stochastic policy gradients in continuous control with deep reinforcement learning using the beta distribution". In: *International conference on machine learning*. PMLR, pp. 834–843.

Coates, Adam, Andrew Ng, and Honglak Lee (2011). "An analysis of single-layer networks in unsupervised feature learning". In: *Proceedings of the fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 215–223.

Consilvio, Alice, Paolo Sanetti, Davide Anguita, Carlo Crovetto, Carlo Dambra, Luca Oneto, Federico Papa, and Nicola Sacco (2019). "Prescriptive maintenance of railway infrastructure: From data analytics to decision support". In: *2019 6th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*. IEEE, pp. 1–10.

Crassidis, John L. and John L. Junkins (2011). *Optimal Estimation of Dynamic Systems, Second Edition (Chapman & Hall/CRC Applied Mathematics & Nonlinear Science)*. 2nd. Chapman & Hall/CRC. ISBN: 1439839859.

Dahal, Keshav P and Nopasit Chakpitak (2007). "Generator maintenance scheduling in power systems using metaheuristic-based hybrid approaches". In: *Electric power systems research* 77.7, pp. 771–779.

Daigle, Matthew and Chetan S Kulkarni (2013). "Electrochemistry-based battery modeling for prognostics". In: *Annual Conference of the PHM Society*. Vol. 5. 1.

Deng, Xiaoheng, Jun Li, Enlu Liu, and Honggang Zhang (2020). "Task allocation algorithm and optimization model on edge collaboration". In: *Journal of Systems Architecture* 110, p. 101778.

Deng, Zui Cha, Jian Ning Yu, and Liu Yang (Apr. 2008). "An inverse problem of determining the implied volatility in option pricing". In: *Journal of Mathematical Analysis and Applications* 340.1, pp. 16–31. ISSN: 0022247X. DOI: 10.1016/j.jmaa.2007.07.075.

Dietterich, Thomas G (2000). "Hierarchical reinforcement learning with the MAXQ value function decomposition". In: *Journal of artificial intelligence research* 13, pp. 227–303.

Doveh, Sivan and Raja Giryes (2019). "DEGAS: Differentiable Efficient Generator Search". In: *arXiv preprint arXiv:1912.00606*.

Du, Yali, Bo Liu, Vincent Moens, Ziqi Liu, Zhicheng Ren, Jun Wang, Xu Chen, and Haifeng Zhang (2021). "Learning Correlated Communication Topology in Multi-Agent Reinforcement learning". In: *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 456–464.

Elsheikh, Ahmed H., Vasily Demyanov, Reza Tavakoli, Mike A. Christie, and Mary F. Wheeler (Jan. 2015). "Calibration of channelized subsurface flow models using nested sampling and soft probabilities". In: *Advances in Water Resources* 75, pp. 14–30. ISSN: 03091708. DOI: 10.1016/j.advwatres.2014.10.006.

Fawzi, Alhussein, Matej Balog, Aja Huang, Thomas Hubert, Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Francisco J R Ruiz, Julian Schrittwieser, Grzegorz Swirszcz, et al. (Oct. 2022). "Discovering faster matrix multiplication algorithms with reinforcement learning". In: *Nature* 610, pp. 47–53. DOI: 10.1038/s41586-022-05172-4.

Feng, Jianghong and Zongrong Gong (2020). "Integrated linguistic entropy weight method and multi-objective programming model for supplier selection and order allocation in a circular economy: A case study". In: *Journal of Cleaner Production* 277, p. 122597.

Feng, Jie, F Richard Yu, Qingqi Pei, Jianbo Du, and Li Zhu (2020). "Joint optimization of radio and computational resources allocation in blockchain-enabled mobile edge computing systems". In: *IEEE Transactions on Wireless Communications* 19.6, pp. 4321–4334.

Fink, Olga, Qin Wang, Markus Svensen, Pierre Dersin, Wan-Jui Lee, and Melanie Ducoffe (2020). "Potential, challenges and future directions for deep learning in prognostics and health management applications". In: *Engineering Applications of Artificial Intelligence* 92, p. 103678.

Finn, Chelsea, Pieter Abbeel, and Sergey Levine (2017). "Model-agnostic meta-learning for fast adaptation of deep networks". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, pp. 1126–1135.

Foerster, Jakob, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson (2018). "Counterfactual multi-agent policy gradients". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. 1.

Foerster, Jakob N, Yannis M Assael, Nando De Freitas, and Shimon Whiteson (2016). "Learning to communicate with deep multi-agent reinforcement learning". In: *arXiv preprint arXiv:1605.06676*.

Foerster, Jakob N, Richard Y Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch (2017). "Learning with opponent-learning awareness". In: *arXiv preprint arXiv:1709.04326*.

Fujimoto, Scott, Herke Hoof, and David Meger (2018). "Addressing Function Approximation Error in Actor-Critic Methods". In: *International Conference on Machine Learning*, pp. 1587–1596.

Gao, Chen, Yunpeng Chen, Si Liu, Zhenxiong Tan, and Shuicheng Yan (2019). "AdversarialNAS: Adversarial Neural Architecture Search for GANs". In: *arXiv preprint arXiv:1912.02037*.

Garrone, Andrea, Simone Minisi, Luca Oneto, Carlo Dambra, Marco Borinato, Paolo Sanetti, Giulia Vignola, Federico Papa, Nadia Mazzino, and Davide Anguita (2023). "Simple Non Regressive Informed Machine Learning Model for Prescriptive Maintenance of Track Circuits in a Subway

Environment". In: *International Conference on System-Integrated Intelligence*. Springer, pp. 74–83.

Glorot, Xavier and Yoshua Bengio (2010). *Understanding the difficulty of training deep feedforward neural networks*. Tech. rep. URL: http://www.iro.umontreal..

Gong, Xinyu, Shiyu Chang, Yifan Jiang, and Zhangyang Wang (2019). "Autogan: Neural architecture search for generative adversarial networks". In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3224–3234.

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio (2014). "Generative adversarial nets". In: *Advances in Neural Information Processing Systems*, pp. 2672–2680.

Gordon, Christopher AK and Efstratios N Pistikopoulos (2022). "Data-driven prescriptive maintenance toward fault-tolerant multiparametric control". In: *AIChE Journal* 68.6, e17489.

Grinblat, Guillermo L, Lucas C Uzal, and Pablo M Granitto (2017). "Class-splitting generative adversarial networks". In: *arXiv preprint arXiv:1709.07359*.

Gulrajani, Ishaan, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville (2017). "Improved training of wasserstein gans". In: *Advances in Neural Information Processing Systems*, pp. 5767–5777.

Guo, Yong, Qi Chen, Jian Chen, Qingyao Wu, Qinfeng Shi, and Mingkui Tan (2019a). "Auto-embedding generative adversarial networks for high resolution image synthesis". In: *IEEE Transactions on Multimedia* 21.11, pp. 2726–2737.

Guo, Yong, Yin Zheng, Mingkui Tan, Qi Chen, Jian Chen, Peilin Zhao, and Junzhou Huang (2019b). "Nat: Neural architecture transformer for accurate and compact architectures". In: *Advances in Neural Information Processing Systems*, pp. 737–748.

Haarnoja, Tuomas, Haoran Tang, Pieter Abbeel, and Sergey Levine (2017). "Reinforcement learning with deep energy-based policies". In: *arXiv preprint arXiv:1702.08165*.

Haarnoja, Tuomas, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. (2018a). "Soft actor-critic algorithms and applications". In: *arXiv preprint arXiv:1812.05905*.

Haarnoja, Tuomas, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. (2018b). "Soft actor-critic algorithms and applications". In: *arXiv preprint arXiv:1812.05905*.

Han, Minghao, Yuan Tian, Lixian Zhang, Jun Wang, and Wei Pan (2019a). "H infinity Model-free Reinforcement Learning with Robust Stability Guarantee". In: *arXiv preprint arXiv:1911.02875*.

Han, Minghao, Yuan Tian, Lixian Zhang, Jun Wang, and Wei Pan (2019b). "H∞ model-free reinforcement learning with robust stability guarantee". In: *arXiv preprint arXiv:1911.02875*.

Han, Minghao, Lixian Zhang, Jun Wang, and Wei Pan (2020). "Actor-critic reinforcement learning for control with stability guarantee". In: *IEEE Robotics and Automation Letters* 5.4, pp. 6217–6224.

Hasselt, Hado (2010). "Double Q-learning". In: *Advances in neural information processing systems* 23, pp. 2613–2621.

He, Hao, Hao Wang, Guang-He Lee, and Yonglong Tian (2019). *Probgan: Towards probabilistic gan with theoretical guarantees*.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2015). "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification". In: *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). "Identity mappings in deep residual networks". In: *European conference on computer vision*. Springer, pp. 630–645.

Henderson, Daniel A., Richard J. Boys, Kim J. Krishnan, Conor Lawless, and Darren J. Wilkinson (Mar. 2009). "Bayesian emulation and calibration of a stochastic computer model of mitochondrial DNA deletions in substantia nigra neurons". In: *Journal of the American Statistical Association* 104.485, pp. 76–87. ISSN: 01621459. DOI: 10.1198/jasa.2009.0005.

Henderson, Peter, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger (2019). *Deep Reinforcement Learning that Matters*. arXiv: 1709.06560 [cs.LG].

Hernandez-Leal, Pablo, Michael Kaisers, Tim Baarslag, and Enrique Munoz de Cote (2017). "A survey of learning in multiagent environments: Dealing with non-stationarity". In: *arXiv preprint arXiv:1707.09183*.

Higdon, Dave, James Gattiker, Brian Williams, and Maria Rightley (2008). "Computer Model Calibration Using High-Dimensional Output". In: *Journal of the American Statistical Association* 103.482, pp. 570–583. ISSN: 01621459. URL: http://www.jstor.org/stable/27640080.

Hoang, Quan, Tu Dinh Nguyen, Trung Le, and Dinh Phung (2018). "MGAN: Training generative adversarial nets with multiple generators". In.

Hu, Xiaosong, Le Xu, Xianke Lin, and Michael Pecht (2020). "Battery lifetime prognostics". In: *Joule* 4.2, pp. 310–346.

Hu, Xiaosong, Changfu Zou, Xiaolin Tang, Teng Liu, and Lin Hu (2019). "Cost-optimal energy management of hybrid electric vehicles using fuel cell/battery health-aware predictive control". In: *ieee transactions on power electronics* 35.1, pp. 382–392.

Huang, Yanjun, Hong Wang, Amir Khajepour, Hongwen He, and Jie Ji (2017). "Model predictive control power management strategies for HEVs: A review". In: *Journal of Power Sources* 341, pp. 91–106.

Hüttenrauch, Maximilian, Sosic Adrian, Gerhard Neumann, et al. (2019). "Deep reinforcement learning for swarm systems". In: *Journal of Machine Learning Research* 20.54, pp. 1–31.

Hwang, W and KS Han (1986). "Cumulative damage models and multi-stress fatigue life prediction". In: *Journal of composite materials* 20.2, pp. 125–153.

Hwangbo, Jemin, Joonho Lee, Alexey Dosovitskiy, Dario Bellicoso, Vassilios Tsounis, Vladlen Koltun, and Marco Hutter (2019). "Learning agile and dynamic motor skills for legged robots". In: *Science Robotics* 4.26, eaau5872.

Hwangbo, Jemin, Inkyu Sa, Roland Siegwart, and Marco Hutter (2017). "Control of a quadrotor with reinforcement learning". In: *IEEE Robotics and Automation Letters* 2.4, pp. 2096–2103.

Ioffe, Sergey and Christian Szegedy (2015). "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *arXiv preprint arXiv:1502.03167*.

Ishii, Koji (2021). "MPC Based Power Allocation for Reliable Wireless Networked Control Systems". In: *IEEE Access* 9, pp. 60913–60922.

Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros (2017). "Image-to-image translation with conditional adversarial networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1125–1134.

Jagtap, Hanumant P, Anand K Bewoor, Ravinder Kumar, Mohammad Hossein Ahmadi, and Lingen Chen (2020). "Performance analysis and availability optimization to improve maintenance schedule for the turbo-generator subsystem of a thermal power plant using particle swarm optimization". In: *Reliability Engineering & System Safety* 204, p. 107130.

Jauhar, Sunil Kumar, Saman Hassanzadeh Amin, and Hossein Zolfagharinia (2021). "A proposed method for third-party reverse logistics partner selection and order allocation in the cellphone industry". In: *Computers & Industrial Engineering* 162, p. 107719.

Jiang, Zhengyao, Dixing Xu, and Jinjun Liang (2017). "A deep reinforcement learning framework for the financial portfolio management problem". In: *arXiv preprint arXiv:1706.10059*.

Jiao, Meng, Dongqing Wang, Yan Yang, and Feng Liu (2021). "More intelligent and robust estimation of battery state-of-charge with an improved regularized extreme learning machine". In: *Engineering Applications of Artificial Intelligence* 104, p. 104407.

Jones, Donald R, Matthias Schonlau, and William J Welch (1998). "Efficient global optimization of expensive black-box functions". In: *Journal of Global optimization* 13.4, pp. 455–492.

Joo, Weonyoung, Wonsung Lee, Sungrae Park, and Il-Chul Moon (2020). "Dirichlet variational autoencoder". In: *Pattern Recognition* 107, p. 107514.

Jordan, Michael I, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul (1999). "An introduction to variational methods for graphical models". In: *Machine learning* 37.2, pp. 183–233.

Julier, Simon J and Jeffrey K Uhlmann (1997). "New extension of the Kalman filter to nonlinear systems". In: *Signal Processing, Sensor Fusion, and Target Recognition VI*. Vol. 3068, p. 182. ISBN: 0819424838. DOI: 10.1117/12.280797.

Kakade, Sham M (2001). "A natural policy gradient". In: *Advances in neural information processing systems* 14, pp. 1531–1538.

Kamandanipour, Keyvan, Mohammad Mahdi Nasiri, Dinçer Konur, and Siamak Haji Yakhchali (2020). "Stochastic data-driven optimization for multi-class dynamic pricing and capacity allocation in the passenger railroad transportation". In: *Expert Systems with Applications* 158, p. 113568.

Kamdar, Renuka, Priyanka Paliwal, and Yogendra Kumar (2018). "A state of art review on various aspects of multi-agent system". In: *Journal of Circuits, Systems and Computers* 27.11, p. 1830006.

Kangasrääsiö, Antti, Jussi P. P. Jokinen, Antti Oulasvirta, Andrew Howes, and Samuel Kaski (June 2019). "Parameter Inference for Computational Cognitive Models with Approximate Bayesian Computation". In: *Cognitive Science* 43.6. ISSN: 0364-0213. DOI: 10.1111/cogs.12738. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cogs.12738.

Kanso, Soha, Mayank Shekhar Jha, and Didier Theilliol (2023). "Degradation Tolerant Optimal Control Design for Linear Discrete-Times Systems". In: *International Conference on Diagnostics of Processes and Systems*. Springer, pp. 398–409.

Kantas, Nikolas, Arnaud Doucet, Sumeetpal S Singh, Jan Maciejowski, and Nicolas Chopin (2015). "On Particle Methods for Parameter Estimation in State-Space Models". In: *Statistical Science* 30.3, pp. 328–351. DOI: 10.1214/14-STS511.

Karras, Tero, Timo Aila, Samuli Laine, and Jaakko Lehtinen (2017). "Progressive growing of gans for improved quality, stability, and variation". In: *arXiv preprint arXiv:1710.10196*.

Karras, Tero, Samuli Laine, and Timo Aila (2019). "A style-based generator architecture for generative adversarial networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4401–4410.

Kennedy, Marc C. and Anthony O'Hagan (2001). "Bayesian calibration of computer models". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 63.3, pp. 425–464. ISSN: 1369-7412. DOI: 10.1111/1467-9868.00294. URL: http://doi.wiley.com/10.1111/1467-9868.00294.

Kingma, Diederik P and Jimmy Ba (2014). "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980*.

Kingma, Diederik P and Jimmy Lei Ba (2015). "Adam: A method for stochastic optimization". In: *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. arXiv: 1412.6980.

Kingma, Diederik P and Max Welling (2014). *Auto-Encoding Variational Bayes*. arXiv: 1312.6114 [stat.ML].

Koch, William, Renato Mancuso, Richard West, and Azer Bestavros (2019). "Reinforcement learning for UAV attitude control". In: *ACM Transactions on Cyber-Physical Systems* 3.2, pp. 1–21.

Konda, Vijay and John Tsitsiklis (1999). "Actor-critic algorithms". In: *Advances in neural information processing systems* 12.

Kotz, Samuel, Narayanaswamy Balakrishnan, and Norman L Johnson (2004). *Continuous multivariate distributions, Volume 1: Models and applications*. Vol. 1. John Wiley & Sons.

Krizhevsky, Alex, Geoffrey Hinton, et al. (2009). "Learning multiple layers of features from tiny images". In.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "Imagenet classification with deep convolutional neural networks". In: *Advances in Neural Information Processing Systems*, pp. 1097–1105.

Kulkarni, Chetan S and Jose Celaya (2019). "Electronics Prognostics". In: *Fault Diagnosis of Dynamic Systems*. Springer, pp. 433–458.

Kulkarni, Chetan S, José R Celaya, Kai Goebel, and Gautam Biswas (2012). "Physics based electrolytic capacitor degradation models for prognostic studies under thermal overstress". In: *PHM Society European Conference*. Vol. 1. 1.

Kumar, Natarajan Chennimalai, Arun K. Subramaniyan, Liping Wang, and Gene Wiggs (Nov. 2013). "Calibrating transient models with multiple responses using bayesian inverse techniques". In: *Proceedings of the ASME Turbo Expo*. Vol. 7 A. American Society of Mechanical Engineers Digital Collection. ISBN: 9780791855263. DOI: 10.1115/GT2013-95857.

Kumar, Vikash, Abhishek Gupta, Emanuel Todorov, and Sergey Levine (2016). "Learning dexterous manipulation policies from experience and imitation". In: *arXiv preprint arXiv:1611.05095*.

Lainé, Julien, Elsa Piollet, Florence Nyssen, and Alain Batailly (2019). "Blackbox optimization for aircraft engine blades with contact interfaces". In: *Journal of Engineering for Gas Turbines and Power* 141.6.

Leng, Jiewu, Qiang Liu, Shide Ye, Jianbo Jing, Yan Wang, Chaoyang Zhang, Ding Zhang, and Xin Chen (2020). "Digital twin-driven rapid reconfiguration of the automated manufacturing system via an open architecture model". In: *Robotics and Computer-Integrated Manufacturing* 63, p. 101895.

Leonori, Stefano, Maurizio Paschero, Fabio Massimo Frattale Mascioli, and Antonello Rizzi (2020). "Optimization strategies for Microgrid energy management systems by Genetic Algorithms". In: *Applied Soft Computing* 86, p. 105903.

Levine, Sergey (2018). "Reinforcement learning and control as probabilistic inference: Tutorial and review". In: *arXiv preprint arXiv:1805.00909*.

Lillicrap, Timothy P, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra (2015). "Continuous control with deep reinforcement learning". In: *arXiv preprint arXiv:1509.02971*.

Littman, Michael L (1994). "Markov games as a framework for multi-agent reinforcement learning". In: *Machine learning proceedings 1994*. Elsevier, pp. 157–163.

Liu, Bin, Jing Lin, Liangwei Zhang, and Uday Kumar (2019a). "A dynamic prescriptive maintenance model considering system aging and degradation". In: *IEEE Access* 7, pp. 94931–94943.

Liu, Chenxi, Liang-Chieh Chen, Florian Schroff, Hartwig Adam, Wei Hua, Alan L Yuille, and Li Fei-Fei (2019b). "Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 82–92.

Liu, Chenxi, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy (2018). "Progressive neural architecture search". In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 19–34.

Liu, Hanxiao, Karen Simonyan, and Yiming Yang (2019c). "DARTS: Differentiable Architecture Search". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=S1eYHoC5FX.

Liu, Shuaiqiang, Anastasia Borovykh, Lech A. Grzelak, and Cornelis W. Oosterlee (Dec. 2019d). "A neural network-based framework for financial model calibration". In: *Journal of Mathematics in Industry* 9.1, pp. 1–28. ISSN: 21905983. DOI: 10.1186/s13362-019-0066-7. arXiv: 1904.10523.

Liu, Xinyang, Zhuoyuan Zheng, İ Esra Büyüktahtakın, Zhi Zhou, and Pingfeng Wang (2021). "Battery asset management with cycle life prognosis". In: *Reliability Engineering & System Safety* 216, p. 107948.

Ljung, Lennart and Svante Gunnarsson (Jan. 1990). "Adaptation and tracking in system identification-A survey". In: *Automatica* 26.1, pp. 7–21. ISSN: 00051098. DOI: 10.1016/0005-1098(90)90154-A.

Lowe, Ryan, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch (2017). "Multi-agent actor-critic for mixed cooperative-competitive environments". In: *arXiv preprint arXiv:1706.02275*.

Maas, Andrew L, Awni Y Hannun, and Andrew Y Ng (2013). "Rectifier nonlinearities improve neural network acoustic models". In: *Proc. icml*. Vol. 30. 1, p. 3.

Mahmood, A Rupam, Dmytro Korenkevych, Brent J Komer, and James Bergstra (2018). "Setting up a reinforcement learning task with a real-world robot". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 4635–4640.

Maia, Ricardo, Jérôme Mendes, Rui Araújo, Marco Silva, and Urbano Nunes (2020). "Regenerative braking system modeling by fuzzy Q-Learning". In: *Engineering Applications of Artificial Intelligence* 93, p. 103712.

Matyas, Kurt, Tanja Nemeth, Klaudia Kovacs, and Robert Glawar (2017). "A procedural approach for realizing prescriptive maintenance planning in manufacturing industries". In: *CIRP Annals* 66.1, pp. 461–464.

Meissner, Robert, Antonia Rahn, and Kai Wicke (2021). "Developing prescriptive maintenance strategies in the aviation industry based on a discrete-event simulation framework for post-prognostics decision making". In: *Reliability Engineering & System Safety* 214, p. 107812.

Meyer, Tobias and Walter Sextro (2014). "Closed-loop control system for the reliability of intelligent mechatronic systems". In: *PHM Society European Conference*. Vol. 2. 1.

Michau, Gabriel and Olga Fink (2021). "Unsupervised transfer learning for anomaly detection: Application to complementary operating condition transfer". In: *Knowledge-Based Systems* 216, p. 106816.

Michau, Gabriel, Yang Hu, Thomas Palmé, and Olga Fink (2020). "Feature learning for fault detection in high-dimensional condition monitoring signals". In: *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability* 234.1, pp. 104–115.

Mirhoseini, Azalia, Anna Goldie, Mustafa Yazgan, Joe Jiang, Ebrahim Songhori, Shen Wang, Young-Joon Lee, Eric Johnson, Omkar Pathak, Sungmin Bae, et al. (2020). "Chip placement with deep reinforcement learning". In: *arXiv preprint arXiv:2004.10746*.

Mirhoseini, Azalia, Anna Goldie, Mustafa Yazgan, Joe Wenjie Jiang, Ebrahim Songhori, Shen Wang, Young-Joon Lee, Eric Johnson, Omkar Pathak, Azade Nazi, et al. (2021). "A graph placement methodology for fast chip design". In: *Nature* 594.7862, pp. 207–212.

Miyato, Takeru, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida (2018). "Spectral normalization for generative adversarial networks". In: *arXiv preprint arXiv:1802.05957*.

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller (2013). "Playing atari with deep reinforcement learning". In: *arXiv preprint arXiv:1312.5602*.

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. (2015). "Human-level control through deep reinforcement learning". In: *nature* 518.7540, pp. 529–533.

Mundhenk, T Nathan, Mikel Landajuela, Ruben Glatt, Claudio P Santiago, Daniel M Faissol, and Brenden K Petersen (2021). "Symbolic regression via neural-guided genetic programming population seeding". In: *arXiv preprint arXiv:2111.00053*.

Nagulapati, Vijay Mohan, Hyunjun Lee, DaWoon Jung, Boris Brigljevic, Yunseok Choi, and Hankwon Lim (2021). "Capacity estimation of batteries: Influence of training dataset size and diversity on data driven prognostic models". In: *Reliability Engineering & System Safety* 216, p. 108048.

Nath, Rahul and Pranab K Muhuri (2021). "Evolutionary Optimization based Solution approaches for Many Objective Reliability-Redundancy Allocation Problem". In: *Reliability Engineering & System Safety*, p. 108190.

Ng, Man-Fai, Jin Zhao, Qingyu Yan, Gareth J Conduit, and Zhi Wei Seh (2020). "Predicting the state of charge and health of batteries using data-driven machine learning". In: *Nature Machine Intelligence*, pp. 1–10.

Nguyen, Tu, Trung Le, Hung Vu, and Dinh Phung (2017). "Dual discriminator generative adversarial nets". In: *Advances in Neural Information Processing Systems*, pp. 2670–2680.

Nian, Rui, Jinfeng Liu, and Biao Huang (2020). "A review on reinforcement learning: Introduction and applications in industrial process control". In: *Computers & Chemical Engineering* 139, p. 106886.

Nicod, Jean-Marc, Brigitte Chebel-Morello, and Christophe Varnier (2017). *From prognostics and health systems management to predictive maintenance 2: knowledge, reliability and decision*. John Wiley & Sons.

OpenAI (2018). *OpenAI Five*. https://blog.openai.com/openai-five/.

Park, Taesung, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu (2019). "Semantic image synthesis with spatially-adaptive normalization". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2337–2346.

Parry, AB, SJ Majumdar, N Peake, and CJ Chapman (1995). "Erosion, corrosion and foreign object damage effects in gas turbines". In: *ROLLS ROYCE PLC-REPORT-PNR*.

Pecht, Michael (2009). "Prognostics and health management of electronics". In: *Encyclopedia of structural health monitoring*.

Peterson, Scott B, JF Whitacre, and Jay Apt (2010). "The economics of using plug-in hybrid electric vehicle battery packs for grid storage". In: *Journal of Power Sources* 195.8, pp. 2377–2384.

Pham, Hieu, Melody Y Guan, Barret Zoph, Quoc V Le, and Jeff Dean (2018). "Efficient neural architecture search via parameter sharing". In: *arXiv preprint arXiv:1802.03268*.

Popp, Timothy, Mehrdad G Shirangi, Ole Petter Nipen, and Anders Berggreen (2020). "Prescriptive data analytics to optimize casing exits". In: *IADC/SPE International Drilling Conference and Exhibition*. OnePetro.

Radford, Alec, Luke Metz, and Soumith Chintala (2015). "Unsupervised representation learning with deep convolutional generative adversarial networks". In: *arXiv preprint arXiv:1511.06434*.

Rakelly, Kate, Aurick Zhou, Deirdre Quillen, Chelsea Finn, and Sergey Levine (2019). "Efficient off-policy meta-reinforcement learning via probabilistic context variables". In: *arXiv preprint arXiv:1903.08254*.

Rashid, Tabish, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson (2018). "Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning". In: *International Conference on Machine Learning*. PMLR, pp. 4295–4304.

Rasmussen, Carl Edward (2003). "Gaussian processes in machine learning". In: *Summer school on machine learning*. Springer, pp. 63–71.

Real, Esteban, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc V Le, and Alexey Kurakin (2017). "Large-scale evolution of image classifiers". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, pp. 2902–2911.

Redutskiy, Yury (2017). "Optimization of safety instrumented system design and maintenance frequency for oil and gas industry processes". In: *Management and Production Engineering Review* 8.

Reed, Scott, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee (2016). "Generative Adversarial Text to Image Synthesis". In: *International Conference on Machine Learning*, pp. 1060–1069.

Richardson, Robert R, Christoph R Birkl, Michael A Osborne, and David A Howey (2018). "Gaussian process regression for in situ capacity estimation of lithium-ion batteries". In: *IEEE Transactions on Industrial Informatics* 15.1, pp. 127–138.

Rokhforoz, Pegah and Olga Fink (2021). "Safe multi-agent deep reinforcement learning for joint bidding and maintenance scheduling of generation units". In: *arXiv preprint arXiv:2112.10459*.

Rokhforoz, Pegah, Blazhe Gjorgiev, Giovanni Sansavini, and Olga Fink (2021). "Multi-agent maintenance scheduling based on the coordination between central operator and decentralized producers in an electricity market". In: *Reliability Engineering & System Safety* 210, p. 107495.

Roychoudhury, I., V. Hafiychuk, and K. Goebel (2013). "Model-based diagnosis and prognosis of a water recycling system". In: *2013 IEEE Aerospace Conference*, pp. 1–9.

Rutter, Carolyn M., Diana L Miglioretti, and James E. Savarino (Dec. 2009). "Bayesian calibration of microsimulation models". English (US). In: *Journal of the American Statistical Association* 104.488, pp. 1338–1350. ISSN: 0162-1459. DOI: 10.1198/jasa.2009.ap07466.

Sabri-Laghaie, Kamyar and Mehdi Karimi-Nasab (2019). "Random search algorithms for redundancy allocation problem of a queuing system with maintenance considerations". In: *Reliability Engineering & System Safety* 185, pp. 144–162.

Sacks, J, W J Welch, J S B Mitchell, and P W Henry (1989). *Design and Experiments of Computer Experiments*. DOI: 10.2307/2245858. URL: http://dx.doi.org/10.2307/2245858.

Sadeghian, Omid, Arman Oshnoei, Saman Nikkhah, and Behnam Mohammadi-Ivatloo (2019). "Multi-objective optimisation of generation maintenance scheduling in restructured power systems based on global criterion method". In: *IET Smart Grid* 2.2, pp. 203–213.

Salimans, Tim, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen (2016). "Improved techniques for training gans". In: *Advances in Neural Information Processing Systems*, pp. 2234–2242.

Samuel, G Giftson and C Christober Asir Rajan (2015). "Hybrid: particle swarm optimization–genetic algorithm and particle swarm optimization–shuffled frog leaping algorithm for long-term generator maintenance scheduling". In: *International Journal of Electrical Power & Energy Systems* 65, pp. 432–442.

Sansó, Bruno, Chris E Forest, and Daniel Zantedeschi (2008). *Inferring Climate System Properties Using a Computer Model*. Tech. rep. 1, pp. 1–38. URL: http://www.ams.ucsc.edu/%7B~%7Ddanielz/.

Schaul, Tom, John Quan, Ioannis Antonoglou, and David Silver (2016). "Prioritized Experience Replay". In: *ICLR (Poster)*.

Schrittwieser, Julian, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. (2020). "Mastering atari, go, chess and shogi by planning with a learned model". In: *Nature* 588.7839, pp. 604–609.

Schulman, John, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz (2015). "Trust region policy optimization". In: *International conference on machine learning*. PMLR, pp. 1889–1897.

Schulman, John, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov (2017). "Proximal policy optimization algorithms". In: *arXiv preprint arXiv:1707.06347*.

Severson, Kristen A, Peter M Attia, Norman Jin, Nicholas Perkins, Benben Jiang, Zi Yang, Michael H Chen, Muratahan Aykol, Patrick K Herring, Dimitrios Fraggedakis, et al. (2019). "Data-driven prediction of battery cycle life before capacity degradation". In: *Nature Energy* 4.5, pp. 383–391.

Shapley, Lloyd S (1953). "Stochastic games". In: *Proceedings of the national academy of sciences* 39.10, pp. 1095–1100.

Shimada, Hayato, Yuichi Kawamoto, and Nei Kato (2021). "Novel Computation and Communication Resources Allocation Using Relay Communications in UAV-mounted Cloudlet Systems". In: *IEEE Transactions on Network Science and Engineering*.

Silver, David, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. (2016). "Mastering the game of Go with deep neural networks and tree search". In: *nature* 529.7587, pp. 484–489.

Silver, David, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller (2014). "Deterministic policy gradient algorithms". In: *International conference on machine learning*. PMLR, pp. 387–395.

Son, Kyunghwan, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi (2019). "Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning". In: *International Conference on Machine Learning*. PMLR, pp. 5887–5896.

Strogatz, Steven H (2018). *Nonlinear dynamics and chaos with student solutions manual: With applications to physics, biology, chemistry, and engineering*. CRC press.

Su, Jianyu, Jing Huang, Stephen Adams, Qing Chang, and Peter A Beling (2022). "Deep multi-agent reinforcement learning for multi-level preventive maintenance in manufacturing systems". In: *Expert Systems with Applications* 192, p. 116323.

Sui, Yu and Shiming Song (2020). "A Multi-Agent Reinforcement Learning Framework for Lithium-ion Battery Scheduling Problems". In: *Energies* 13.8, p. 1982.

Sun, Guofeng, Zhiqiang Tian, Renhua Liu, Yun Jing, and Yawen Ma (2020). "Research on coordination and optimization of order allocation and delivery route planning in take-out system". In: *Mathematical Problems in Engineering* 2020.

Sunehag, Peter, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, et al. (2017). "Value-decomposition networks for cooperative multi-agent learning". In: *arXiv preprint arXiv:1706.05296*.

Sutton, Richard S, Andrew G Barto, et al. (1998). *Introduction to reinforcement learning*. Vol. 135. MIT press Cambridge.

Sutton, Richard S and Andrew G Barto (2018). *Reinforcement learning: An introduction*. MIT press.

Sutton, Richard S, Andrew G Barto, and Ronald J Williams (1992). "Reinforcement learning is direct adaptive optimal control". In: *IEEE Control Systems Magazine* 12.2, pp. 19–22.

Tan, Ming (1993). "Multi-agent reinforcement learning: Independent vs. cooperative agents". In: *Proceedings of the tenth international conference on machine learning*, pp. 330–337.

Taylor, Annalisa T, Thomas A Berrueta, and Todd D Murphey (2021). "Active learning in robotics: A review of control principles". In: *Mechatronics* 77, p. 102576.

Tesauro, Gerald, Nicholas K Jong, Rajarshi Das, and Mohamed N Bennani (2006). "A hybrid reinforcement learning approach to autonomic resource allocation". In: *2006 IEEE International Conference on Autonomic Computing*. IEEE, pp. 65–73.

Tian, Yuan, Manuel Arias Chao, Chetan Kulkarni, Kai Goebel, and Olga Fink (2022a). "Real-time model calibration with deep reinforcement learning". In: *Mechanical Systems and Signal Processing* 165, p. 108284.

Tian, Yuan, Minghao Han, Chetan Kulkarni, and Olga Fink (2022b). "A prescriptive Dirichlet power allocation policy with deep reinforcement learning". In: *Reliability Engineering & System Safety* 224, p. 108529.

Tian, Yuan, Klaus-Rudolf Kladny, Qin Wang, Zhiwu Huang, and Olga Fink (2023). "Multi-agent actor-critic with time dynamical opponent model". In: *Neurocomputing* 517, pp. 165–172.

Tian, Yuan, Qin Wang, Zhiwu Huang, Wen Li, Dengxin Dai, Minghao Yang, Jun Wang, and Olga Fink (2020a). "Off-policy reinforcement learning for efficient and effective gan architecture search". In: *European Conference on Computer Vision*. Springer, pp. 175–192.

Tian, Zheng, Ying Wen, Zhichen Gong, Faiz Punakkath, Shihao Zou, and Jun Wang (2019). "A regularized opponent model with maximum entropy objective". In: *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pp. 602–608.

Tian, Zheng, Shihao Zou, Ian Davies, Tim Warr, Lisheng Wu, Haitham Bou Ammar, and Jun Wang (2020b). "Learning to communicate implicitly by actions". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 05, pp. 7261–7268.

Tran, Ngoc-Trung, Tuan-Anh Bui, and Ngai-Man Cheung (2018). "Dist-gan: An improved gan using distance constraints". In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 370–385.

Turner, Ryan and Carl Edward Rasmussen (2010). "Model based learning of sigma points in unscented Kalman filtering". In: *Proceedings of the 2010 IEEE International Workshop on Machine Learning for Signal Processing, MLSP 2010*, pp. 178–183. ISBN: 9781424478774. DOI: 10.1109/MLSP.2010.5589003.

Ulyanov, Dmitry, Andrea Vedaldi, and Victor Lempitsky (2016). "Instance normalization: The missing ingredient for fast stylization". In: *arXiv preprint arXiv:1607.08022*.

Unagar, Ajaykumar, Yuan Tian, Manuel Arias Chao, and Olga Fink (2021). "Learning to Calibrate Battery Models in Real-Time with Deep Reinforcement Learning". In: *Energies* 14.5, p. 1361.

Urban, Louis A (1973). "Gas path analysis applied to turbine engine condition monitoring". In: *Journal of Aircraft* 10.7, pp. 400–406.

Van Hasselt, Hado, Arthur Guez, and David Silver (2016). "Deep reinforcement learning with double q-learning". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 30. 1.

Vater, Johannes, Lars Harscheidt, and Alois Knoll (2019). "Smart manufacturing with prescriptive analytics". In: *2019 8th International Conference on Industrial Technology and Management (ICITM)*. IEEE, pp. 224–228.

Vinyals, Oriol, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. (2019). "Grandmaster level in StarCraft II using multi-agent reinforcement learning". In: *Nature* 575.7782, pp. 350–354.

Volkanovski, Andrija, Borut Mavko, Tome Boševski, Anton Čauševski, and Marko Čepin (2008). "Genetic algorithm optimisation of the maintenance scheduling of generating units in a power system". In: *Reliability Engineering & System Safety* 93.6, pp. 779–789.

Wang, Hanchao and Jun Huan (2019). "Agan: Towards automated design of generative adversarial networks". In: *arXiv preprint arXiv:1906.11080*.

Wang, Qin, Gabriel Michau, and Olga Fink (2019a). "Domain adaptive transfer learning for fault diagnosis". In: *2019 Prognostics and System Health Management Conference (PHM-Paris)*. IEEE, pp. 279–285.

Wang, Qin, Cees Taal, and Olga Fink (2021). "Integrating expert knowledge with domain adaptation for unsupervised fault diagnosis". In: *IEEE Transactions on Instrumentation and Measurement* 71, pp. 1–12.

Wang, Ting-Chun, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro (2018). "High-resolution image synthesis and semantic manipulation with conditional gans". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8798–8807.

Wang, Wei, Mingqiang Lin, Yongnian Fu, Xiaoping Luo, and Hanghang Chen (2020). "Multi-objective optimization of reliability-redundancy allocation problem for multi-type production systems considering redundancy strategies". In: *Reliability Engineering & System Safety* 193, p. 106681.

Wang, Wei, Yuan Sun, and Saman Halgamuge (2019b). "Improving MMD-GAN Training with Repulsive Loss Function". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=HygjqjR9Km.

Wang, Yue, Xiaohua Zeng, Dafeng Song, and Nannan Yang (2019c). "Optimal rule design methodology for energy management strategy of a power-split hybrid electric bus". In: *Energy* 185, pp. 1086–1099.

Wang, Yujie, Zhendong Sun, and Zonghai Chen (2019d). "Development of energy management system based on a rule-based power distribution strategy for hybrid power sources". In: *Energy* 175, pp. 1055–1066.

Warde-Farley, David and Yoshua Bengio (2016). "Improving generative adversarial networks with denoising feature matching". In.

Wasserman, Larry (2013). *All of statistics: a concise course in statistical inference*. Springer Science & Business Media.

Watkins, Christopher JCH and Peter Dayan (1992). "Q-learning". In: *Machine learning* 8.3-4, pp. 279–292.

Wei, Ermo, Drew Wicke, David Freelan, and Sean Luke (2018). "Multiagent soft q-learning". In: *2018 AAAI Spring Symposium Series*.

Wen, Ying, Yaodong Yang, Rui Luo, Jun Wang, and Wei Pan (2018). "Probabilistic Recursive Reasoning for Multi-Agent Reinforcement Learning". In: *International Conference on Learning Representations*.

Wen, Ying, Yaodong Yang, Rui Luo, Jun Wang, and Wei Pan (2019). "Probabilistic recursive reasoning for multi-agent reinforcement learning". In: *arXiv preprint arXiv:1901.09207*.

Williams, Ronald J (1992). "Simple statistical gradient-following algorithms for connectionist reinforcement learning". In: *Machine learning* 8.3-4, pp. 229–256.

Xiao, Lei, Sanling Song, Xiaohui Chen, and David W Coit (2016). "Joint optimization of production scheduling and machine group preventive maintenance". In: *Reliability Engineering & System Safety* 146, pp. 68–78.

Xie, Sirui, Hehui Zheng, Chunxiao Liu, and Liang Lin (2018). "SNAS: stochastic neural architecture search". In: *arXiv preprint arXiv:1812.09926*.

Xie, Zhaoming, Patrick Clary, Jeremy Dao, Pedro Morais, Jonathan Hurst, and Michiel van de Panne (2019). "Iterative Reinforcement Learning Based Design of Dynamic Locomotion Skills for Cassie". In: *arXiv preprint arXiv:1903.09537*.

Xiong, Rui, Jiayi Cao, and Quanqing Yu (2018). "Reinforcement learning-based real-time power management for hybrid energy storage system in the plug-in hybrid electric vehicle". In: *Applied energy* 211, pp. 538–548.

Xu, Tao, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He (2018). "Attngan: Fine-grained text to image generation with attentional generative adversarial networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1316–1324.

Xu, Xiaodong, Shengjin Tang, Chuanqiang Yu, Jian Xie, Xuebing Han, and Minggao Ouyang (2021a). "Remaining Useful Life Prediction of Lithium-ion Batteries Based on Wiener Process Under Time-Varying Temperature Condition". In: *Reliability Engineering & System Safety* 214, p. 107675.

Xu, Yue, Dechang Pi, Shengxiang Yang, and Yang Chen (2021b). "A novel discrete bat algorithm for heterogeneous redundancy allocation of multi-state systems subject to probabilistic common-cause failure". In: *Reliability Engineering & System Safety* 208, p. 107338.

Yang, Duo, Xu Zhang, Rui Pan, Yujie Wang, and Zonghai Chen (2018a). "A novel Gaussian process regression model for state-of-health estimation of lithium-ion battery using charging curve". In: *Journal of Power Sources* 384, pp. 387–395.

Yang, Jianwei, Anitha Kannan, Dhruv Batra, and Devi Parikh (2017). "Lr-gan: Layered recursive generative adversarial networks for image generation". In: *arXiv preprint arXiv:1703.01560*.

Yang, Tianyu, Yulin Hu, M Cenk Gursoy, Anke Schmeink, and Rudolf Mathar (2018b). "Deep reinforcement learning based resource allocation in low latency edge computing networks". In: *2018 15th International Symposium on Wireless Communication Systems (ISWCS)*. IEEE, pp. 1–5.

Yang, Yaodong, Rui Luo, Minne Li, Ming Zhou, Weinan Zhang, and Jun Wang (2018c). "Mean field multi-agent reinforcement learning". In: *International Conference on Machine Learning*. PMLR, pp. 5571–5580.

Yang, Yaodong, Ying Wen, Jun Wang, Liheng Chen, Kun Shao, David Mguni, and Weinan Zhang (2020). "Multi-agent determinantal q-learning". In: *International Conference on Machine Learning*. PMLR, pp. 10757–10766.

Ye, Hao, Geoffrey Ye Li, and Biing-Hwang Fred Juang (2019). "Deep reinforcement learning based resource allocation for V2V communications". In: *IEEE Transactions on Vehicular Technology* 68.4, pp. 3163–3173.

Yin, Yanling, Xuhui Bu, Panpan Zhu, and Wei Qian (2022). "Point-to-Point Consensus Tracking Control for Unknown Nonlinear Multi-Agent Systems Using Data-Driven Iterative Learning". In: *Neurocomputing*.

Yin, Yilin and Song-Yul Choe (2020). "Actively temperature controlled health-aware fast charging method for lithium-ion battery using nonlinear model predictive control". In: *Applied Energy* 271, p. 115232.

Yuan, Shuo, Chengpu Yu, and Ping Wang (2022). "Suboptimal Linear Quadratic Tracking Control for Multi-Agent Systems". In: *Neurocomputing*.

Zhang, Han, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena (2019). "Self-Attention Generative Adversarial Networks". In: *International Conference on Machine Learning*, pp. 7354–7363.

Zhang, Han, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas (2017). "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks". In: *Proceedings of the IEEE international conference on computer vision*, pp. 5907–5915.

Zhang, Hanxiao and Yan-Fu Li (2021). "Robust optimization on redundancy allocation problems in multi-state and continuous-state series-parallel systems". In: *Reliability Engineering & System Safety*, p. 108134.

Zhang, Kaiqing, Zhuoran Yang, and Tamer Başar (2021a). "Multi-agent reinforcement learning: A selective overview of theories and algorithms". In: *Handbook of Reinforcement Learning and Control*, pp. 321–384.

Zhang, Liang, Hao Zheng, Tao Wan, Donghan Shi, Ling Lyu, and Guowei Cai (2021b). "An integrated control algorithm of power distribution for islanded microgrid based on improved virtual synchronous generator". In: *IET Renewable Power Generation*.

Zhang, Shuo, Rui Xiong, and Jiayi Cao (2016). "Battery durability and longevity based power management for plug-in hybrid electric vehicle with hybrid energy storage system". In: *Applied Energy* 179, pp. 316–328.

Zhang, Tianhao, Yueheng Li, Chen Wang, Guangming Xie, and Zongqing Lu (2021c). "Fop: Factorizing optimal joint policy of maximum-entropy multi-agent reinforcement learning". In: *International Conference on Machine Learning*. PMLR, pp. 12491–12500.

Zhang, Xi, Haicheng Tu, Jianbo Guo, Shicong Ma, Zhen Li, Yongxiang Xia, and Chi Kong Tse (2021d). "Braess paradox and double-loop optimization method to enhance power grid resilience". In: *Reliability Engineering & System Safety* 215, p. 107913.

Zhang, Xiaoxiong, Song Ding, Bingfeng Ge, Boyuan Xia, and Witold Pedrycz (2021e). "Resource allocation among multiple targets for a defender-attacker game with false targets consideration". In: *Reliability Engineering & System Safety* 211, p. 107617.

Zhang, Y., L. Guo, B. Gao, T. Qu, and H. Chen (2020). "Deterministic Promotion Reinforcement Learning Applied to Longitudinal Velocity Control for Automated Vehicles". In: *IEEE Transactions on Vehicular Technology* 69.1, pp. 338–348.

Zheng, Fangdan, Jiuchun Jiang, Martha A Zaidan, Wei He, and Michael Pecht (2015). "Prognostics of lithium-ion batteries using a deterministic Bayesian approach". In: *2015 IEEE Conference on Prognostics and Health Management (PHM)*. IEEE, pp. 1–4.

Zheng, Yan, Zhaopeng Meng, Jianye Hao, Zongzhang Zhang, Tianpei Yang, and Changjie Fan (2018). "A deep bayesian policy reuse approach against non-stationary agents". In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 962–972.

Zhong, Zhao, Junjie Yan, Wei Wu, Jing Shao, and Cheng-Lin Liu (2018). "Practical block-wise neural network architecture generation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2423–2432.

Ziebart, Brian D (2010). "Modeling purposeful adaptive behavior with the principle of maximum causal entropy". In.

Zoph, Barret and Quoc V Le (2017). "Neural architecture search with reinforcement learning". In: *International Conference on Learning Representations*.