General Reinforcement Learning Agents for Crop Management

Version of October 30, 2023

Athanasios Theocharis

General Reinforcement Learning Agents for Crop Management

THESIS

submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE

by

Athanasios Theocharis

Interactive Intelligence Group
Department of Intelligent Systems
Faculty EEMCS, Delft University of Technology
Delft, the Netherlands
`www.ewi.tudelft.nl`

ETH Zürich
Universitätstrasse 6,
8092 Zürich, Switzerland
`https://inf.ethz.ch/`

Author: Athanasios Theocharis
Student ID: 5621771

Abstract

Agriculture plays a vital role in the global economy, providing the necessary food and resources for human survival. With the world's population projected to surge, the demand for food is set to escalate in the coming decades. This increasing demand, coupled with the challenges posed by climate change and the detrimental effects of pollution due to fertilizers, underscores the urgency for more efficient and sustainable crop management strategies. Effective crop management is a complex and time-consuming task that involves various factors, including climate conditions and soil quality. Traditional crop management strategies often rely on expert knowledge to guide the decision-making process, which may be suboptimal and prone to error. Reinforcement learning (RL) has gained significant attention in recent years as a promising approach for decision-making and control in agriculture, aiding in the management process.

RL environments such as CyclesGym [51], accommodate the design of agents that operate within an agricultural system, often surpassing the performance of traditional strategies. However, the optimal policy may vary heavily depending on the specific field location, due to its specific weather conditions and soil quality. In this thesis, we aim to investigate the use of RL for managing fields in multiple locations with the aim of reducing training time and data and increasing robustness compared to independent training. To this end, we plan to use multi-task learning methods and optimizers to reduce total training time, to improve RL agents' adaptability to changing environments, and to reduce data usage required for maximum performance across multiple agricultural fields.

Acknowledgements

First and foremost, I would like to express my deepest gratitude to Prof. Dr. Frans Oliehoek. His willingness to accept me as his master's student and guide me through this journey has been invaluable. His continuous support and guidance, and the constructive feedback in our every meeting have been integral for the completion of this thesis. I am equally indebted to Dr. Matteo Turchetta, my daily supervisor, who helped me shape my research mindset, supporting me with weekly meetings, providing constructive feedback to my choices and direction throughout this endeavour. Prof. Dr. Andreas Krause hosted me in ETH Zurich and in his group, Learning & Adaptive Systems, giving me the opportunity for this thesis, for which I am extremely thankful. I would also like to express my appreciation to Prof. Dr. J. W. Böhmer, who was kind enough to join the thesis committee and provide valuable insights on the project.

However, the completion of this thesis would not have been possible without the unwaivering support of those dear to to me. Ariwor, Tzotzo, Konstantinos, Sandal, Koz, Sam, Malech - I am more than lucky to have you by my side throughout the past year. You were the ones that inspired me to embark on this journey in the first place, and the ones that stuck with me in my highs and lows.

Lastly, I want to extend my deepest appreciation to my family, that supported me every step of the way - I know that it was not the most straightforward path, and I am deeply grateful for your understanding.

Athanasios Theocharis
Delft, the Netherlands
October 30, 2023

Contents

List of Figures

Acronyms

CGM Crop Growth Model. vii, 16, 18, 19

DRL Deep Reinforcement Learning. 9

MDP Markovian Decision Process. v, 7

MTRL Multi-Task Reinforcement Learning. v, vi, viii, 4–6, 9, 10, 24–27, 31–39, 41, 43

N Nitrogen. 24

PPO Proximal Policy Optimization. 9

RL Reinforcement Learning. v, viii, 2, 4, 6–9, 15, 17–20, 22, 24, 25, 29, 33, 35–39

Chapter 1

Introduction

1.1 Motivation

The agricultural sector is facing multiple challenges in the recent years. The global population is projected to increase by 25% until 2050 [2], shown in Figure 1.1, leading to an analogous rise in food demand. Concurrently, the need for consistent and sustainable crop yields is harder to fulfill as the world is experiencing more frequent and severe weather events due to climate change [18]. The production and excessive use of fertilizers such as Nitrogen (N) contribute to greenhouse gas emissions [34] and environmental pollution[33]. Such use, alone, is more than the aviation and shipping industry combined, as shown in 1.2, emphasizing the need for optimized and environmentally-conscious crop management strategies.

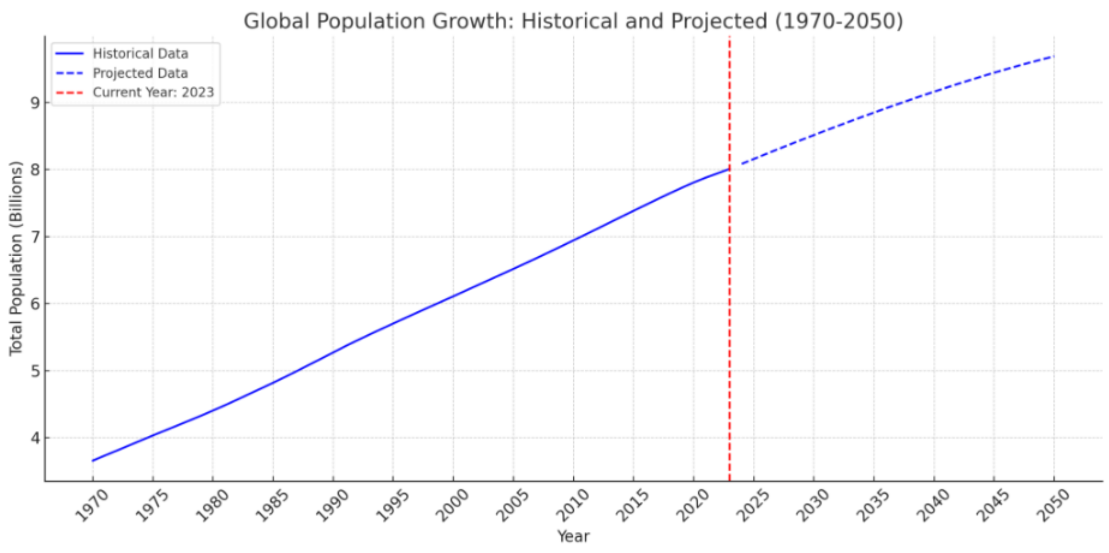

Figure 1.1: Global population size projection. As population increases, the global food demand increases as well. Plot based on UN data [2].

Domain expertise and advancements in agronomic research have proven effective in maintaining and enhancing crop yields, leveraging time-tested methods and domain expertise. However, there are inherent limitations to such practices, as they lack adaptability to events such as sudden weather events, and are mainly based on general empirical guidelines [15]. As the problem of optimizing crop management is highly complex, due to the multitude of variables that can affect plant growth and can be controlled there exists potential to further harness efficiencies through intelligent control and decision-making systems [54]. Reinforcement Learning

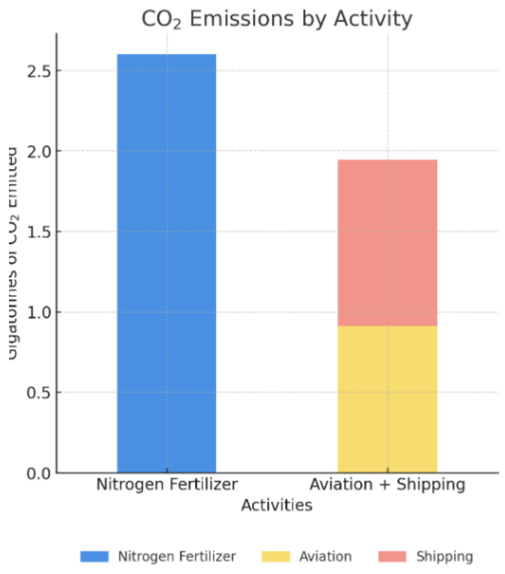

Figure 1.2: Environmental impact of Nitrogen as fertilizer is bigger than aviation combined with shipping. Plot based on data from [34] [33].

(RL), with its inherent ability to learn optimal action sequences and adapt to evolving conditions, presents a promising avenue for optimization[31]. Analogous to human learning from trial and error, RL agents can navigate the complexities of agricultural environments, aligning interventions such as irrigation, fertilization or crop rotations in an optimal sequence for maximizing yield and sustainability [51]. Based on previous literature [51] it is shown that an RL agent can outperform the current employed strategies by 11-33%, translating to hundreds of millions of dollars lost, when scaled to continent size. See in Figure 1.3.

However, the application of RL in agriculture currently faces a significant challenge: the need to train a single agent for each crop field, as an agent cannot perform optimally even in geographically close locations, as shown in Figure 1.4 and Table 1.1. This approach is not only computationally intensive but lacks scalability, limiting the broader deployment of Reinforcement Learning agents across diverse agricultural lands.

The objective of this thesis is to explore and address this limitation. By utilizing the Cycles-Gym [51] framework, this project seeks to develop general RL agents that can adapt to different crop fields, optimizing crop management across a range of conditions. While progress in RL training efficiency will reduce required resources, such agents can also provide more robust and

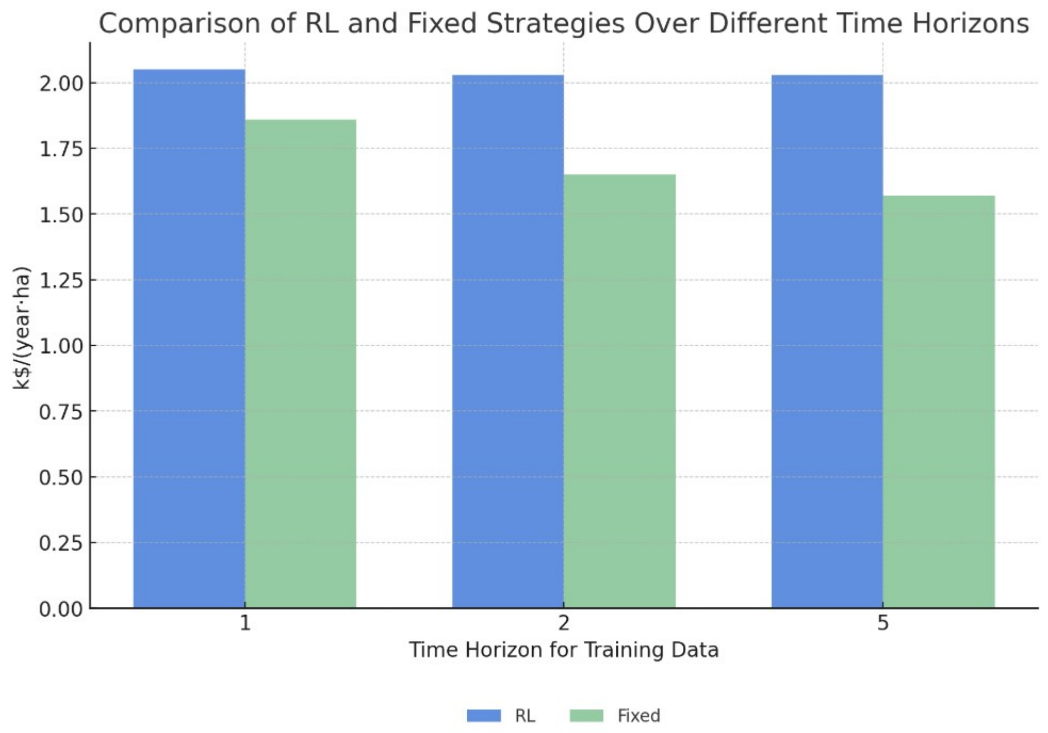

Figure 1.3: Comparison of rewards, in k$ per hectare, between a Reinforcement Learning Policy and a Fixed Policy. A Fixed Policy is a farmer's pre-decided strategy, that is based on empirical guidelines. Rewards are measured based on economic profit, per year, per hectare of land. The x-axis shows the time horizon of the training data, thus comparing different scenarios. Rewards are averaged for each year in the specified time period. Plot derived from [51]

adaptable models in the long run. In doing so, this work aims to contribute to the growing need for scalable and efficient agricultural practices that can meet the demands of a growing population and address the pressing challenges posed by pollution and climate change-induced agricultural vulnerabilities.

1.1.1 Single field training limitations

Using a single agent for each individual field might seem like a simple solution, but it's not feasible for broader applications. This raises essential questions: How do soil element changes and weather variations affect the performance of an agent? How extensive can an area be for a single agent to effectively manage? To that end, we trained four distinct agents, each for a specific location in Switzerland: Trelex, Senarclen, Orges, and Bassins. These locations are spaced between 5 and 25 km apart. The performance of each agent across these locations is presented in Table 1.1 and Figure 1.4. Interestingly, even for locations A,B just 5 km apart, an agent trained on data of location A, can significantly underperform compared to an agent specifically trained in location B. This highlights the difficulty in transferring knowledge between even closely situated areas.

Considering the broader context, one can calculate the total computational time needed for Europe. Training a single agent for a specific agricultural area takes about 2–3 hours. Europe has a vast agricultural expanse of 1.57 million km^2. Dividing this area into 25 km^2 tiles (keeping in mind that even a 5 km difference can lead to inferior agent performance), one can grasp the scale of the challenge ahead.

$$\begin{aligned}\text{Training time per field} &: 2\text{–}3\,\text{hours}\\ \text{Total area of farmland in Europe} &: 1.57\,\text{million km}^2\\ \text{Total training time for Europe} &: \frac{1,570,000 \times 2.5\,\text{hours}}{25\ \text{km}^2}\\ &\approx 157\,\text{thousand hours}\\ &\approx 18\ \text{years}\end{aligned}$$

Naturally, spending 18 years of computational time to train $\approx$ 65000 models, needing the respective amount of data and human resources, is an ineffective way to optimize crop management. By scaling this world-wide, and taking into account the organizational and infrastructural effort, it becomes evident that both pre-determined, empirical-based policies that are currently employed, and single-agent-per-field strategies are quite impractical. Instead the proposed approach aims to develop general RL agents capable of optimal decision making across diverse fields, leading to saving computational costs, increasing crop yield, and reducing the total climate impact of agriculture.

Table 1.1: Rewards of Reinforcement Learning agents evaluated in multiple swiss fields in k$ per hectare of land. While all of these fields are in a radius of 12km, the rewards differ significantly.

		Deployed in			
		Trelex	**Senarclen**	**Orges**	**Bassins**
Trained in	**Trelex**	0.72	0.45	-10	0.42
	Senarclen	-12	0.58	-9.7	-15
	Orges	0.46	0.41	0.49	0.42
	Bassins	-11	0.45	-9.9	0.32

1.2 Proposed Approach

The approach taken in this research to address the challenges of optimizing crop management across locations using Reinforcement Learning is outlined in this section. The proposed methods center around Multi-Task Reinforcement Learning (MTRL) and optimizer techniques.

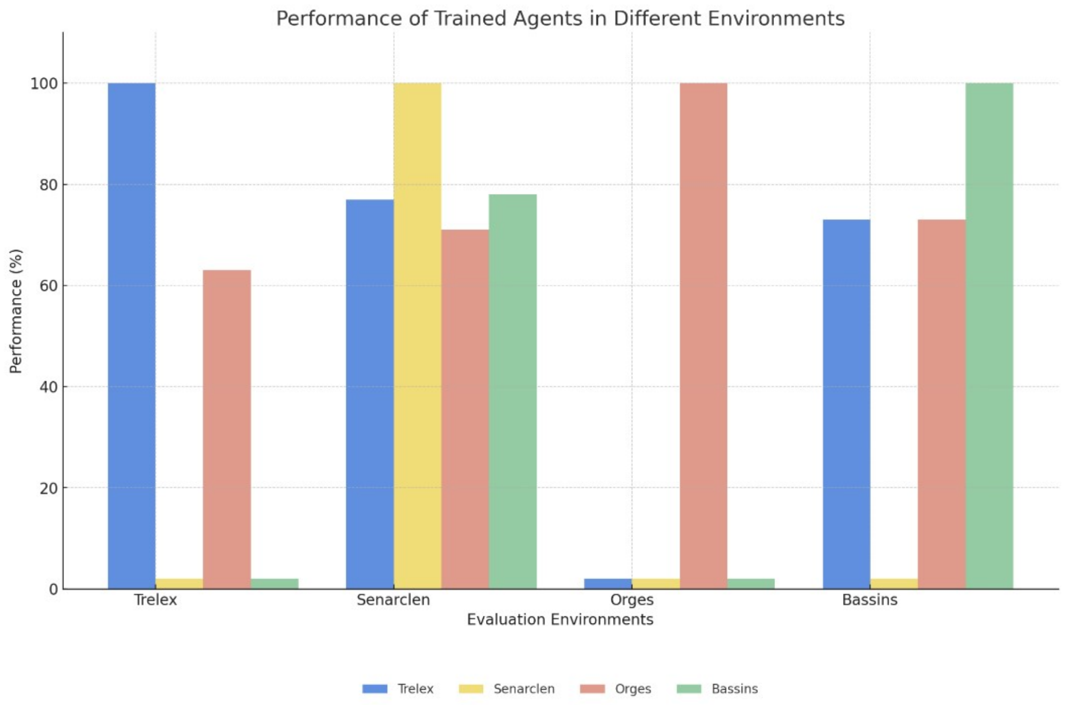

Figure 1.4: Performance comparison of multiple agents in different locations. In this plot, the metric used is Performance % $= \frac{p_B^A}{p_B^B} \times 100$, where p_B^A is the performance of an agent trained in location A and evaluated in location B. The distance between the 4 locations shown in this barplot are 5-25km. Note: There should be negative bars, given the Table 1.1, however, to save space in this plot that serves as motivation, any performance % ≤ 0, is given as 0, to show that inability to provide any meaningful rewards.

1.2.1 Multi-Task Reinforcement Learning (MTRL)

Multi-Task Reinforcement Learning provides a framework that allows a single agent to learn multiple tasks simultaneously. By sharing common features and leveraging the intrinsic relationship among different tasks, MTRL promotes more efficient learning and facilitates knowledge transfer across different tasks[13]. This provides the capability of agent generalization across diverse fields, varying in soil, weather condtions and suitable crops. The significance of this extends beyond computational efficiency; in an industry in which actors can only target for narrow profit margins [25], small percentage gains in crop yield can translate into substantial economic impact. Therefore, the choice of MTRL in this research is based on the complexity and requirements of modern agricultural optimization, highlighting its importance for scalable and sustainable methods. In this research, sequential task sampling [29] is the main applied MTRL method.

1.2.2 Multi-Task Learning Optimizers

Effective training of MTRL models requires specialized optimization techniques. Various optimizers have been developed to enhance the convergence and stability of MTRL training. In this research, the focus is on PCGrad [59], that tries to solve the conflicting gradients problem during MTRL [59], based on the hypothesis that diverse agricultural fields, with their respective data, lead to conflicting gradients during RL training.

1.2.3 Conclusion of the Proposed Approach

This study provides the first Multi-Task Reinforcement Learning approach for crop management. Our proposed approach combines the principles of Multi-Task Reinforcement Learning and MTRL-specific optimizers' techniques to create a robust and scalable solution to the intricate problem of optimizing crop management across diverse fields and conditions.

1.3 Research Objectives

In the context of using Reinforcement Learning in agriculture, this section defines our research objectives. We focus on understanding the advantages and limitations generalized RL agents for agricultural deployment. More specifically:

1. *How can Reinforcement Learning agents be designed to generalize across diverse agricultural areas, without the need for field-specific training?*
2. *How do general Reinforcement Learning agents compare to field-specific agents in terms of training speed, performance, and adaptability?*
3. *What methods offer the greatest efficiency in terms of computational resources and time in developing reinforcement learning agents capable of managing diverse agricultural areas?*
4. *What is the largest area that can be managed by a single Reinforcement Learning (RL) agent?*

Chapter 2

Background

2.1 Markovian Decision Process (MDP)

Markov Decision Processes (MDPs) provide a mathematical framework for modeling sequential decision-making, especially under uncertainty, which, in turn, is a core concept for Reinforcement Learning (RL) [4]. Formally, an MDP is defined by a tuple (S, A, P, R), where:

- S is a finite set of states,
- A is a finite set of actions,
- P is the state transition probability function, and
- R is the reward function.

A visualization of these components is shown in 2.1

The state transition probability function $P : S \times A \times S \to [0,1]$ defines the probability of transitioning from state s to state s' after taking action a, and is given by:

$$P(s'|s,a) = \Pr(S_{t+1} = s'|S_t = s, A_t = a) \tag{2.1}$$

where S_t and A_t denote the state and action at time t, respectively. The reward function

$$R : S \times A \times S \to R$$

provides the expected reward received after transitioning from state s to state s' due to action a, and is defined as:

$$R(s,a,s') = E[R_{t+1}|S_t = s, A_t = a, S_{t+1} = s'] \tag{2.2}$$

,

A policy $\pi : S \times A \to [0,1]$ specifies the probability of choosing action a in state s. The objective in an MDP is to find an optimal policy π^* that maximizes the expected cumulative reward over time. This is often expressed in terms of the value function $V^{\pi}(s)$ or the action-value function $Q^{\pi}(s,a)$, which are defined as:

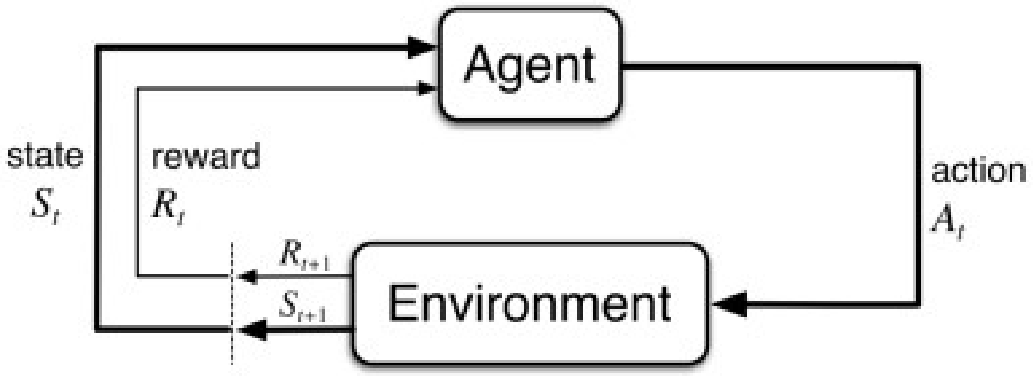

Figure 2.1: Visualization of interactions between agents and environment in MDP. Image taken from [48]

$$V^{\pi}(s) = E_{\pi}\left[\sum_{t=0}^{\infty}\gamma^t R_t | S_0 = s\right] \tag{2.3}$$

,

$$Q^{\pi}(s,a) = E_{\pi}\left[\sum_{t=0}^{\infty}\gamma^t R_t | S_0 = s, A_0 = a\right] \tag{2.4}$$

,

where $0 \leq \gamma < 1$ is the discount factor which models the agent's consideration for future rewards. The optimal policy π^* can then be obtained by solving the Bellman optimality equations:

$$V^*(s) = \max_a \left[R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a)V^*(s')\right] \tag{2.5}$$

,

$$Q^*(s,a) = R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) \max_{a'} Q^*(s',a') \tag{2.6}$$

.

MDPs are foundational for understanding the dynamics in which Reinforcement Learning (RL) operates. They provide a systematic way of modeling and solving decision-making problems in multiple domains by translating real-world problems into a formalized computational structure [40].

2.2 Reinforcement Learning

Reinforcement Learning is a broader computational approach to learning optimal decision-making in uncertain environments. Unlike MDPs, RL does not assume prior knowledge of transition probabilities or reward functions. Instead, an agent learns from the feedback—rewards and new states—it receives upon taking actions in different states, progressively improving its policy through a process of trial-and-error [48]. This learning paradigm allows the agent to learn either the underlying MDP's structure or a policy that performs well without necessarily learning the MDP's parameters [24].

The fundamental goal in RL is to learn an optimal policy that maximizes the expected cumulative reward over time. This is achieved through an ongoing process where the agent continually interacts with the environment, receiving feedback in the form of rewards that inform the agent about the quality of the actions taken. The agent uses this feedback to update its policy, aiming to improve its future performance.

RL encompasses a wide array of algorithms designed to efficiently explore the action space, balance the trade-off between exploration (trying new actions to discover their effects) and exploitation (choosing actions that are known to yield good rewards), and learn a policy either by estimating the value functions or by directly optimizing the policy. Algorithms such as Q-learning, SARSA, and Temporal Difference Learning are foundational methods in RL that estimate the value of state-action pairs to guide the policy towards higher rewards [48].

2.2.1 Deep Reinforcement Learning (DRL)

Recently, Deep Reinforcement Learning (DRL), which combines deep neural networks with RL, has significantly expanded the capabilities of traditional RL [35]. By tackling complex, high-dimensional problems, DRL overcomes the limitations of classical RL methods, extending its applicability to real-world challenges like autonomous driving, robotics, and game playing [5].

Proximal Policy Optimization (PPO)

Proximal Policy Optimization (PPO) stands as a common algorithm in DRL applications, aiming for a balance between Value Iteration and Policy Iteration to ensure stable and robust learning [43]. Unlike traditional policy optimization methods, PPO focuses on a moderate step towards the new policy, without extreme policy changes. This moderation is achieved through a surrogate objective function that penalizes changes in the policy that are "too far" from the old policy. The objective function for PPO is articulated as:

$$L(\theta) = E_t\left[\min\left(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t\right)\right] \tag{2.7}$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ denotes the probability ratio, $\hat{A}_t$ is the estimated advantage at time t, and ε is a hyperparameter controlling the clipping. The clipping function ensures the updates remain bounded, promoting stable and effective learning.

2.2.2 Multi-Task Reinforcement Learning

Multi-Task Reinforcement Learning (MTRL) extends the conventional Reinforcement Learning (RL) framework to handle multiple tasks simultaneously. This extension is particularly relevant in real-world scenarios where a system is required to perform several tasks, each with potentially differing objectives and with differing input data. The core idea in MTRL is to leverage the intrinsically common features and shared structures between the different tasks to improve the learning efficiency and performance on each task[10]. This is achieved by sharing representations, value functions, or policies across different tasks. [42]

Multi-Task Reinforcement Learning (MTRL) relies on the benefits of utilizing relations across tasks to improve the performance on any single task, specifically with context-based representations[46]. Furthermore, addressing the challenge of parameter sharing among different tasks, parameter-compositional MTRL can train a single policy that can be applied to a set of different tasks [47].

To achieve high performance across tasks and overcome challenges associated with MTRL, multiple architectures and methodologies have been proposed. Based on attention-mechanisms [52], an attention-based mixture model manages the shared information across different tasks and addresses the challenge of how parameters in the network should be reused across tasks[12]. Additionally, a survey on Multi-Task Deep Reinforcement Learning discusses state-of-the-art approaches by comparing and contrasting recent solutions such as DISTRAL (DIStill TRAnsfer Learning) [50], IMPALA (Importance Weighted Actor-Learner Architecture)[14], and PopArt[21], elaborating on the research challenges associated with multitask rl [53]. Moreover, the concept of soft modularization is introduced to optimize the learning process, by dynamically reconfiguring the base policy network for each task [57].

Sequential Task Sampling

The Multi-Task Reinforcement Learning training in this research, is using sequential task sampling [23] [29] [28], where model weights are updated sequentially, based on the rollouts of each task. Given a set of T tasks, a MTRL optimization is performed from an initialization ϕ. For each task t = 1,..., $\mathcal{T}$, we sample minibatches $\mathcal{B}_t^{(1)}, \ldots, \mathcal{B}_t^{(K)}$ from task data $\mathcal{D}_t$ and change the gradients based on them, as shown in the following equations:

$$\theta_t^{(0)} = \phi, \tag{2.8}$$

$$\theta_t^{(k)} = \theta_t^{(k-1)} - \alpha \frac{\partial \mathcal{L}\left(\theta_t^{(k-1)}; \mathcal{B}_t^{(k)}\right)}{\partial \theta_t^{(k-1)}} \tag{2.9}$$

2.2.3 Optimizers

One common approach in Multi-Task Learning is to optimize an objective function, which is typically a weighted average of the task-specific objective functions, namely unitary scalarization [28]. This approach aims at finding a balance between the tasks so that the learning process is beneficial to all tasks involved. However, it's often debated whether the conventional methods of simply optimizing a weighted average of the task losses are effective [55]. Research has led to the proposal of specialized optimization algorithms tailored for deep multi-task models. These algorithms are believed to yield solutions superior to those obtained by merely optimizing a weighted average of the task losses[59].

PCGrad

A method in the multi-task optimizer category is PCGrad, which aims to solve the challenge of conflicting gradients during training [59], as shown in 2.2. Vectors **x** and **y** similarity is defined as $\cos(\mathbf{x},\mathbf{y})$. for the cosine similarity between vectors. If the following two conditions hold:

- conflicting gradient directions: $\cos\left(\nabla_{\theta_{\|}}\mathcal{L}_i, \nabla_{\theta_{\|}}\mathcal{L}_j\right) < 0$ for some $i, j \in \mathcal{T}$
- differing gradient magnitudes: $\left\|\nabla_{\theta_{\|}}\mathcal{L}_i\right\| \gg \left\|\nabla_{\theta_{\|}}\mathcal{L}_j\right\|$ for some $i, j \in \mathcal{T}$

learning is slowed and possibly ineffective [59]. To solve such conflicts, a projection of the per-task gradient onto the gradients' normal plane. If g is the update direction and $[\mathbf{x}]_+ := \max(\mathbf{x},\mathbf{0})$. Given per-task gradients $\nabla_{\theta_{\|}}\mathcal{L}_i$, PCGrad [59] projects each task gradient onto the normal plane of all the gradients with which it conflicts:

$$\left[\mathbf{g}_i \leftarrow \nabla_{\theta_{\|}}\mathcal{L}_i, \mathbf{g}_i \leftarrow \mathbf{g}_i + \left[\frac{-\mathbf{g}_i^T \nabla_{\theta_{\|}}\mathcal{L}_j(\mathbf{x})}{\left\|\nabla_{\theta_{\|}}\mathcal{L}_j\right\|^2}\right]_+ \nabla_{\theta_{\|}}\mathcal{L}_j \forall j \in \mathcal{T}\backslash\{i\}\right] \forall i \in \mathcal{T},$$

$$\mathbf{g} = -\sum_{i\in\mathcal{T}} \mathbf{g}_i,$$

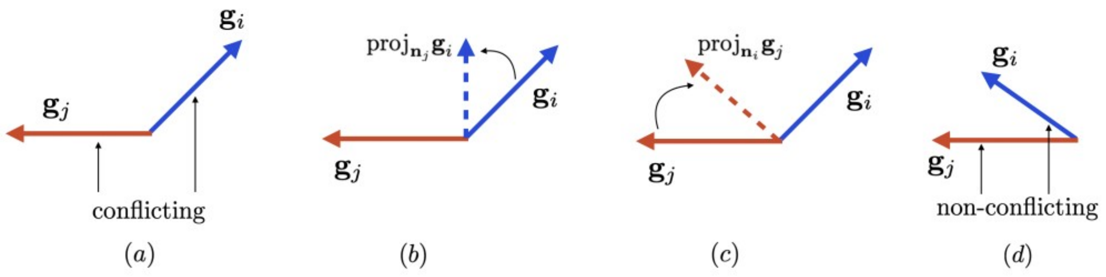

Figure 2.2: Conflicting and non-conflicting gradients in PCGrad, and their projections [59]

2.3 Crop Management Optimization in Agriculture

Crop Management Optimization (CMO) is a domain within agricultural research aimed at enhancing productivity and sustainability. It involves optimizing the allocation of resources and timing of agricultural activities to maximize yield, minimize costs, and ensure environmental sustainability[20]. The application of optimization techniques in agriculture facilitates better decision-making amid uncertain environmental conditions and limited resources.

The integration of machine learning (ML) and artificial intelligence (AI) in CMO has shown potential in advancing optimized agricultural practices. Computational techniques such as ML and AI enable the analysis of large datasets, which include weather data, soil conditions, and crop performance, to generate insights and predictions. Through the application of ML and AI, more informed decision-making regarding planting, irrigation, fertilization, and pest control can be achieved[30].

Reinforcement Learning (RL), a subset of ML, has found application in CMO. By modeling the agricultural decision-making process as a Markov Decision Process, RL can be employed to learn optimal policies in agricultural tasks. The iterative learning process inherent in RL allows for the continuous improvement of policies based on feedback from the environment, making it a dynamic tool to address the evolving challenges faced in agricultural management[11].

2.3.1 Machine Learning for Crop Management

Machine Learning (ML) has emerged as a potent tool for addressing complex problems in various domains, including agriculture. In crop management, ML can be employed to analyze large datasets derived from various sources such as satellite imagery, weather stations, and soil sensors to generate actionable insights[30]. Through ML algorithms, patterns and relationships within data can be discerned, enabling predictive modeling and decision support for various aspects of crop management including pest and disease detection, yield prediction, and resource optimization.

The application of ML in crop management has been facilitated by the advent of big data technologies and the increasing availability of high-resolution spatial and temporal data. Remote sensing technologies, for instance, provide a continuous stream of data regarding crop health, soil conditions, and environmental factors, which can be analyzed using ML to monitor crop conditions and predict yields[30]. Moreover, ML can be employed to optimize irrigation schedules, fertilization plans, and other resource allocations, thereby contributing to the overall efficiency and sustainability of agricultural practices.

Various machine learning algorithms have been adapted to tackle distinct challenges within crop management. For instance, supervised learning algorithms have been employed for tasks such as disease detection and yield prediction where labeled data is available[6]. In scenarios where labeled data are scarce or expensive to obtain, unsupervised and semi-supervised learning paradigms are beneficial. They find use in plant systems biology for tasks like clustering, dimensionality reduction, self-supervised learning, and transfer learning[56]. On the other hand, Reinforcement Learning (RL) is another paradigm that finds application in optimizing decision-making processes in crop management. [16]

2.3.2 Reinforcement Learning in Agriculture

Reinforcement Learning (RL) aligns well with the challenges and requirements of crop management optimization. The inherent nature of RL to learn from interactions with the environment and improve decision-making over time makes it apt for agricultural settings where the decisions pertaining to irrigation, fertilization, and pest control have a significant impact on the yield. Through continuous learning and adaptation, RL can help in designing dynamic policies that can better cope with the uncertainties and variabilities in agricultural scenarios[16].

One of the platforms facilitating the application of RL in agriculture is DSSATGym, which serves as a bridge between RL research and realistic crop management tasks. DSSATGym provides an open-source RL environment interfaced with the Decision Support System for Agrotechnology Transfer (DSSAT), a high fidelity crop simulator. This setup allows for the simulation of real-world sequential decision problems in agriculture, aiding in the development and testing of RL algorithms for crop management optimization[54].

CyclesGym is another platform that leverages the increasing abundance of data in agricultural systems to design adaptive policies through RL. The platform encourages learning long-term crop management strategies, as shown in 2.3 by adapting to environmental conditions, providing a natural technique for learning policies in sequential decision-making problems within agriculture[51]. In this research, specifically, we performed experiments using Cycles and CyclesGym, as it is the only package that allows for multi-year modelling of crop growth.

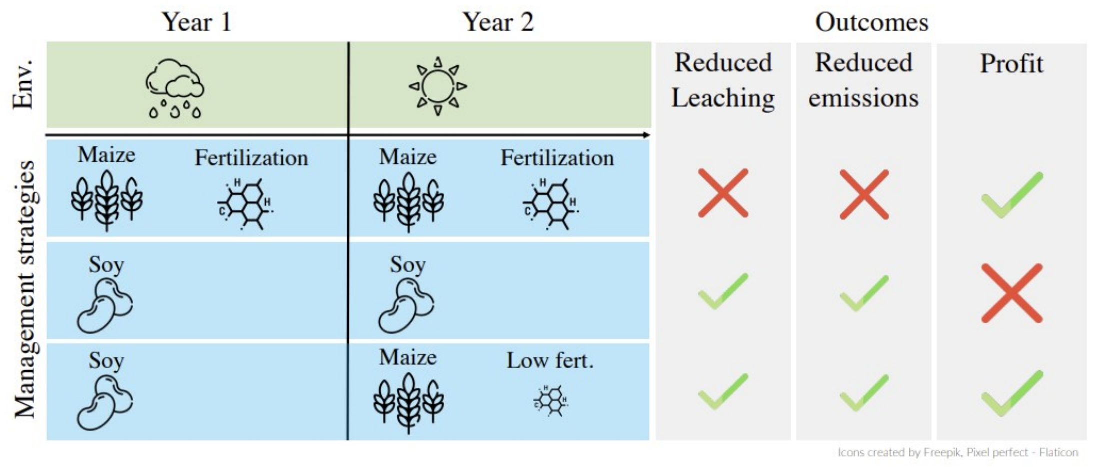

Figure 2.3: Agricultural management for long-term decisions. Multi-year strategies affect both environmental and economic impact. [51]

FarmGym aims to foster both the RL research and interaction between agronomy and RL communities. It models a farm as a dynamical system with many interacting entities and takes a gamification approach to make the environment engaging yet realistic for testing and developing RL algorithms for stochastic and dynamic farm management scenarios[31].

CropGym provides a configurable environment for conducting RL research in crop management. Built around PCSE, a python library with various crop simulation models, CropGym facilitates learning fertilization management policies using process-based crop growth models. It particularly focuses on reducing the environmental impact of nitrogen fertilizers by optimizing fertilizer management strategies through RL[37].

Chapter 3

Methods

3.1 Environment Description

Understanding the environment in which a Reinforcement Learning agent operates is crucial for designing effective RL approaches. The environment serves as the source of the agent's input, and the agent interacts with the environment by taking actions. Consequently, the performance of the RL agent is largely influenced by its understanding of the environment. For this reason, it is essential to explain the environment from both the Reinforcement Learning (RL) and physical perspectives. By defining the environment comprehensively, we can better address the problems and challenges involved in applying RL to crop management.

3.1.1 Physical Problem

The physical problem addressed in this research is specific to crop management with an emphasis on optimizing nitrogen application and irrigation for maximal crop growth and profit while minimizing environmental impacts. Both nitrogen and water are required resources for crop development, yet their mismanagement can lead to undesirable consequences [32]. Excessive nitrogen application can result in soil degradation, water pollution, and greenhouse gas emissions, whereas inefficient irrigation practices can waste water, degrade soil, and contribute to groundwater depletion [7].

The challenge arises from the dynamic and interconnected factors that influence crop growth, nitrogen uptake, and water use, including daily weather conditions, soil properties, and agronomic practices [45]. Weather variables like temperature, precipitation, and solar radiation directly impact crop growth and water demand, while soil properties such as clay, sand, organic matter content, and bulk density affect nutrient and water availability and retention [27].

Achieving optimal nitrogen and irrigation management in this complex system requires a data-driven approach that can account for the interaction of multiple factors. By building upon a Reinforcement Learning (RL) framework that incorporates detailed weather and soil data with realistic simulations, the research seeks to help farmers make informed choices for sustainable and profitable crop management across diverse fields, improving crop yield and quality while reducing environmental impacts.

Data Description

The weather information used in the simulations includes daily weather data over an area, downloaded from AGERA5 [8] and Cycles [27]. Each daily weather record comprises of several variables: *YEAR* and *DOY*, representing the year and numerical day of the year; *PP*, precipitation in mm/day; *TX* and *TN*, the maximum and minimum daily temperatures in degrees Celsius; *SOLAR*, the daily solar radiation in MJ/m^2/day; *RHX* and *RHN*, the maximum and minimum relative humidity in % ; and *WIND*, the average wind speed in m/s.

Soil information is also used in simulations. The soil data includes properties of different soil layers, with columns indicating *LAYER*, *THICK*, *CLAY*, *SAND*, *ORGANIC*, *BD*, *FC*, *PWP*, *SON*, *NO3*, *NH4*, *BYP_H*, and *BYP_V*. *LAYER* indicates the layer number, and *THICK* is the thickness of each soil layer in meters. *CLAY* and *SAND* represent the clay and sand particle size fractions of each layer in %. *ORGANIC* is the organic matter concentration in %. *BD* is bulk density in Mg/m^3, and *FC* and *PWP* are the field capacity and permanent wilting point volumetric water contents, respectively, in m^3/m^3. *SON* is the soil organic nitrogen mass in kg/ha for each soil layer. *NO3* and *NH4* are the nitrate and ammonium masses in kg/ha for each soil layer. *BYP_H* and *BYP_V* represent the fractions of horizontal and vertical bypass flows in each soil layer.

3.2 Cycles

Cycles is a Crop Growth Model (CGM) [58] for soil, water, and plant simulations, developed to understand and predict biogeochemical processes in agroecosystems. Designed to operate at various temporal and spatial scales, Cycles facilitates the study of complex interactions between soil, water, and plants in agricultural systems. The model incorporates the effects of weather, soil properties, crop management practices, and plant growth on carbon and nitrogen dynamics, soil water flow, and nutrient cycling. Cycles allows users to explore and evaluate various agricultural management practices and their implications for soil health, crop yield, and environmental sustainability. The model integrates daily weather data, detailed soil profile information, and crop-specific parameters to simulate plant growth, nutrient uptake, and soil carbon and nitrogen dynamics.

3.2.1 Running Cycles Simulations

Cycles performs simulations by using parameters and limits defined in multiple different data files. Given a set of files as input, it provides a set of files as output, containing detailed information regarding the simulation. In more detail:

- The set of **input** files contains information regarding the simulation hyperparameters, operations (such as fertilization, irrigation, planting, tilling, and harvesting) applied during the simulations, and soil/weather data as described before.

- The set of **output** files contains information regarding daily environmental data, crop growth, biomass accumulation, nitrogen effects and deposits, water balance and more.

More information can be found in Appendix A.

A simple single-year simulation overview of the operations affecting the results is shown in Figure 3.1 providing the basic operations that affect the it. Specific information, such as amount of fertilizer, day of the year to perform an operation are given by **input** files.

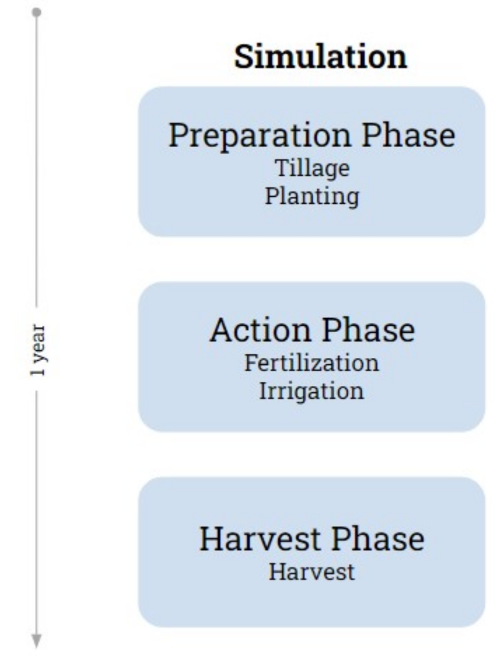

Figure 3.1: Cycles simulation overview

3.3 CyclesGym

3.3.1 Introduction to CyclesGym

CyclesGym [51] is a Reinforcement Learning environment designed for long-term planning in agroecosystems. It is based on the multi-year, multi-crop Crop Growth Model (CGM) Cycles [27] and provides an OpenAI Gym [9] compatible environment for simulating and optimizing Reinforcement Learning (RL) crop management policies in Python. **Why CyclesGym?**

- **Long-Term Planning**: Enables the modeling of multi-year environments with multiple crops, reflecting the real-world complexities of agricultural management.
- **Modularity**: Offers a modular OpenAI gym [9] environment, with a fully customizable set of state space, reward functions, weather generators using Python's inheritance functionality.
- **Crop options**: Offers an extensive selection of ready crop configurations, while also allowing custom-made crops.

- **Real-World Benchmark**: Serves as a novel benchmark for researchers, bridging the gap between theoretical RL and practical agricultural applications.

Cycles Integration in CyclesGym

CyclesGym is utilizing mechanistic crop growth models (CGMs), namely the Cycles [27] model, to simulate crop growth and soil conditions driven by the weather conditions and farming operations.

It is important to understand why there is a need to use such a mechanistic model. The Cycles model is integrated into CyclesGym to provide a realistic representation of agricultural ecosystems, simulating crop growth processes. In RL, to solve such a complex application requires a simulator that can reproduce real-world conditions [17]. While datasets can supply information on daily weather patterns and initial soil conditions, they often fall short in capturing the nuanced effects of specific fertilization and irrigation operations, with regard to amount and time of the year, on both soil and plant conditions. This is where the Cycles CGM contributes the required data. By simulating the effects of each agricultural operation, CyclesGym ensures that the RL agent receives precise feedback during both training and evaluation phases, enabling more informed and effective decision-making in crop management. Naturally, an alternative option is to gather data using equipment from real-world fields, based on the actions of our agent, but it is easily understandable that this is an extremely expensive, inefficient and lengthy process. Hence, the integration of Cycles with a gym environment is the only computationally efficient way of modeling and training RL agents for crop management.

As shown in Figure 3.2 the RL Environment is using the simulation data provided by the mechanistic Crop Growth Model in S_{t+1}, based on the actions A_t
However, using mechanistic models comes with disadvantages.

- **Dependency** on the quality of the model. The performance of an RL agent is directly affected by how good a mechanistic model is to simulate real-word.
- **Sim2Real gap** The simulations generated by a Crop Growth Model and reality can differ greatly. This can be tackled in a certain degree by either calibrating the model [3]

Environment Configuration in CyclesGym

- **State Space**: The observation space includes variables provided by the Cycles model, such as soil moisture, nutrient levels, crop growth stage and daily weather conditions provided by a dataset.
- **Actions**: Actions encompass various management techniques, reflecting real-world agricultural practices, such as irrigating, fertilizing and.
- **Rewards**: Rewards are based on the final economic profit of the farmer after harvesting - or multiple harvests in multi-year experiments.

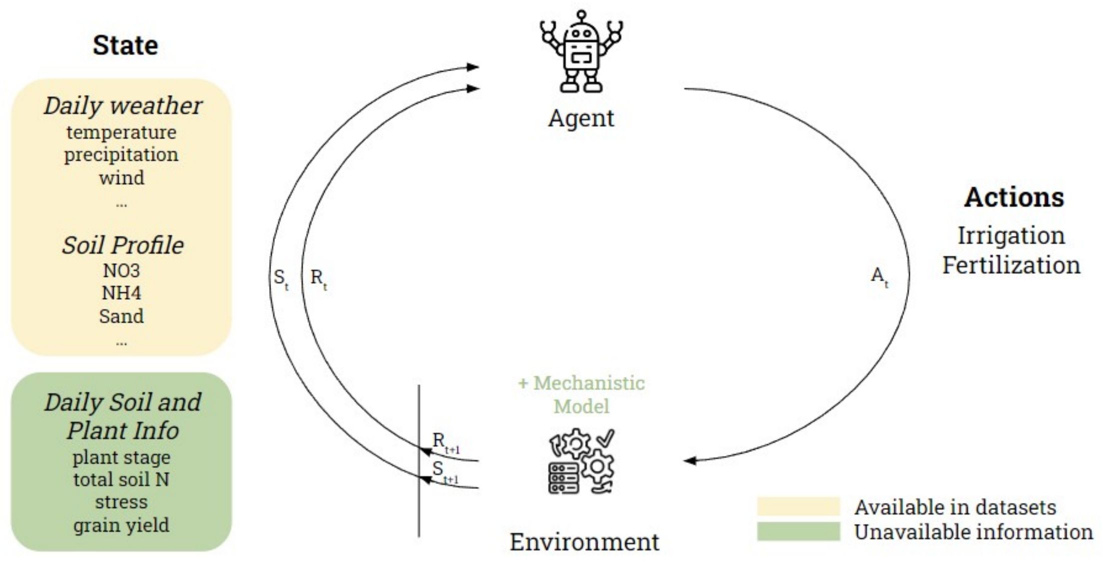

Figure 3.2: CyclesGym training loop and integration of Cycles Crop Growth Model

Training and Evaluation with CyclesGym CyclesGym serves as the primary environment for training and evaluating RL agents in this research. The Cycles model plays a crucial role in simulating underlying agroecosystem dynamics, enabling the learning of adaptive strategies that outperform expert practices in various scenarios.

1. **Environment Initialization**: Configuring CyclesGym with specific crop fields, weather scenarios, and management objectives.

2. **Agent Interaction**: The RL agent interacts with the CyclesGym environment, guided by the Cycles model's simulations, taking actions, observing new states, and receiving rewards.

3. **Evaluation and Analysis**: Assessing the RL agent's performance in unseen test scenarios, reflecting the real-world applicability of the learned strategies.

Agent Interaction with CyclesGym The RL agent's interaction with the CyclesGym environment is a critical phase where the Cycles model plays a vital role. This interaction consists of the following components (refer to Figure 3.2):

- **Action Execution**: The RL agent takes actions such as irrigation and fertilization. These actions are then processed by the Cycles model, which simulates their effects on the soil and crop growth dynamics.

- **State Observation**: After executing an action, the RL agent observes the new state of the environment. The Cycles model provides some these state observations, reflecting the real-world effects of the agent's actions on variables like soil moisture, nutrient levels, and crop growth stage, alongside daily weather conditions provided by the dataset.

- **Reward Calculation**: The reward function in CyclesGym guides the RL agent's learning. The Cycles model contributes to defining the reward by providing information such as crop yield, nitrogen leaching, etc.

Conclusion The integration of the Cycles model into CyclesGym provides a robust and realistic platform for exploring long-term crop management strategies. By simulating complex interactions between crops, soil, weather, and management practices, CyclesGym facilitates the development of RL agents capable of addressing modern agriculture's complex challenges.

3.4 Action Space

The action space, which determines the set of decisions an RL agent can make, primarily comprises two critical agricultural interventions: fertilizing and irrigation. By optimizing these actions, we aim to enhance the agent's ability to adapt and perform across varied land types and weather conditions.

3.4.1 Action Space for Fertilization-only Experiments

In the fertilization-only scenario, the action space is defined by the amount of nitrogen fertilizer applied to the crops. In the equation 3.4.1, representing the action space, each unit corresponds to an application of 40 kg of nitrogen fertilizer per hectare of land.

$$A_t = \{k \cdot 40\,\text{kg/ha}\,\text{N fertilizer} \,|\, k \in \{0,1,2,3,4\}\}$$

(3.1)

The 40 kg amount was chosen as it was already used in previous research [49].

3.4.2 Action Space for Fertilization and Irrigation Experiments

In the scenario with both fertilization and irrigation actions, the action space is defined by the amount of nitrogen fertilizer and the amount of irrigation water applied to the crops. In the equation 3.2, each unit corresponds to an application of 40 kg/ha of nitrogen fertilizer and 6 m^3/ha of irrigation water.

$$A_{N,I} = \{k \cdot 40\,\text{kg/ha}\,\text{N fertilizer}, k \cdot 6\,\text{m}^3/\text{ha}\,\text{Irrigation water} \,|\, k \in \{0,1,2,3,4\}\} \tag{3.2}$$

The 6m$\hat{3}$ water amount was chosen as it was already used in previous research [49].

3.5 State Space

While in Markov Decision Processes (MDPs) [40], states are assumed to be Markovian, containing relevant information regarding the past and the future, in practice, and in agriculture, access to such data is rare and expensive [51]. To address this, Partially Observable Markov Decision Processes (POMDPs) [36] are used by CyclesGym.

In our application, we use a set of values, comprised of the values in each observation in Table 3.1 to represent the each state in the POMDP.

Observation	**Description**	**Optional**
PP	Daily Precipitation	No
TX	Maximum daily temperature	No
TN	Minimum daily temperature	No
SOLAR	Solar radiation	No
RHX	Maximum daily relative humidity	No
RHN	Minimum daily relative humidity	No
STAGE	Stage in the plant life cycle	No
CUM. BIOMASS	Cumulative plant biomass	No
N STRESS	Daily N stress value of the crop	No
WATER STRESS	Daily water stress value of the crop	No
ORG SOIL N	Sum of microbial biomass N and stabilized soil organic N pools	No
PROF SOIL NO3	Soil profile nitrate-N content	No
PROF SOIL NH4	Soil profile ammonium-N content	No
N TO DATE	Nitrogen amount used in fertilization so far in the ismulation	No
WATER TO DATE	Water amount used in irrigation during the simulation so far	No
Y	Years left to simulate	Yes
TASK	Task Identifier	Yes

Table 3.1: Observation variables and their descriptions.

3.6 Reward Functions

3.6.1 Financial-Based Reward Function

In the initial design of our experiments, the reward function $R(s_t, a_t)$ considered only the yield amount and nitrogen use. It was defined as follows:

$$R(s_t, a_t) = \begin{cases} Y_t - C_N \cdot N_t & \text{if } t \text{ is harvest day} \\ -C_N \cdot N_t & \text{otherwise} \end{cases} \tag{3.3}$$

where:

- Y_t is the yield amount at time t

- N_t is the amount of nitrogen used at time t
- C_N is the cost of nitrogen per unit, given as 4.96 USD/kg of N

3.6.2 Integrated Financial and Environmental Reward Function

The updated reward function $R(s_t, a_t)$ considers yield amount, nitrogen use, leaching fee, irrigation fee, and balance fee. It can be defined as follows:

$$R(s_t, a_t) = \begin{cases} Y_t - C_N \cdot N_t - L_t \cdot F_L - I_t \cdot F_I - B_t \cdot I_{\{B_t \geq 10\}} \cdot F_B & \text{if } t \text{ is harvest day} \\ -C_N \cdot N_t - L_t \cdot F_L - I_t \cdot F_I & \text{otherwise} \end{cases} \tag{3.4}$$

where:

- Y_t is the yield amount at time t.
- N_t is the amount of nitrogen used at time t.
- C_N is the cost of nitrogen per unit, given as 4.96 USD/kg N.
- L_t is the amount of nitrogen leached at time t.
- F_L is the leaching fee, given as 2.0 USD/kg N/ha.
- I_t is the amount of irrigation water used at time t.
- F_I is the irrigation fee, given as 1.1 USD/mm/ha.
- B_t is the nitrogen balance at time t, calculated as $N_{\text{fertilizer}} - N_{\text{removed}}$ where $N_{\text{fertilizer}}$ is the N applied as mineral fertilizer, and N_{removed} is the N harvested in maize grain, calculated from crop yield and an estimated grain N concentration of 11.5 g N/kg grain as in Eagle et al. (2020).
- $I_{\{B_t \geq 10\}}$ is an indicator function that is 1 if $B_t \geq 10$ and 0 otherwise.
- F_B is the balance fee, given as 2 USD.

3.7 Dataset

In this research, a dataset was created to facilitate the exploration of RL approaches in crop management, and to systematically study the effects of the proposed methods in a controlled environment. The dataset comprises of weather and soil data from various locations, as well as modifications to existing weather files and Corn-specific information.

3.7.1 Data Collection

Weather Data

Weather data were collected from multiple locations, including Leon (Spain), Chaiyaphum (Thailand), Bremen (Germany), Dresden (Germany), Stuttgart (Germany), Munich (Germany), and Munster (Germany). The data were sourced from the AGERA5 dataset [8], which provides global weather data tailored for agriculture. AGERA5 is available from the Earth Informatics and can be accessed at `https://www.earthinformatics.eu/tools/agera5-global-weather-data-set-tailored-agriculture`. Cycles ships with weather files on New Holland and Rock Springs, both located in USA [1].

Soil Data

Soil data were collected from the SoilGrid [39] dataset, which provides global predictions for soil properties and classes at a fine spatial resolution. SoilGrid is available from the ISRIC - World Soil Information and can be accessed at `https://soilgrids.org`. As a go-to file, the GenericHagerstown soil file that was shipped alongside Cycles, was used as well.

3.7.2 Weather File Generation

Based on the Rock Springs and New Holland weather files, which can be found included in the Cycles release [1], new weather files were generated by modifying specific elements. Two different sets of modifications were performed:

- The daily maximum temperature was changed from -10 to +18, in steps of 2, resulting in 15 new weather files for each of the Rock Springs and New Holland weather files, codenamed `RSTX1`, `RSTX2`...`RSTX15`, and `NHTX1`, `NHTX2`...`NHTX15`. `RS` stands for Rock Springs based files, `NH` stands for New Holland based files and `TX` stands for max temperature.

- Four elements were modified by changing their daily values according to the following ranges:

  - Maximum temperature (tx): -5 to 12, in steps of 2
  - Maximum relative humidity (rhx): -20 to 21, in steps of 15
  - Minimum relative humidity (rhn): -20 to 21, in steps of 15
  - Wind speed (wind): 0 to 9, in steps of 4

  This resulted in 243 new weather files for each of the Rock Springs and New Holland weather files, codenamed `RS1`, `RS2`...`RS243`, and `NH1`, `NH2`...`NH243`. `RS` stands for Rock Springs based files, and `NH` stands for New Holland based files.

3.7.3 Corn-Specific Information Modification

Cycles ships with a set of plant-specific information, that is used as input in simulations. As before, this set was extended and corn-specific information was modified by changing the following parameters:

- Flowering range: 800 to 1500, in steps of 100
- Maturity range: 1500 to 2500, in steps of 100

These modifications allowed for the exploration of the effects of different flowering and maturity ranges on crop growth and management strategies.
According to experts, the configuration `CornRM.110` was the one that was more suitable for the fields of Central Europe, and as such, it was selected during the RL experiments more often.

3.7.4 Conclusion

The creation of this dataset provides a rich and diverse set of data for training and evaluating RL agents in crop management. The modifications to weather files and Corn-specific information enabled the systematic exploration of various scenarios, the development of adaptive policies for multiple environmental conditions and the evaluation of the impact of method choices on the RL agent performance.

3.8 Task Selection

In this research, task selection was vital for evaluating MTRL methods in crop management. The objective was to develop RL agents capable of managing diverse field conditions. Therefore, tasks that are addressed by similar policies were not ideal for evaluating the effectiveness of an MTRL method. The primary step involved identifying tasks from the dataset that necessitated distinct policies for effective resolution, ensuring a comprehensive evaluation of multi-task learning.

As **task**, we define the set of input files used for a Cycles simulation, and specifically, the set of weather, soil and crop files.

3.8.1 Policy Evaluation

To identify diverse tasks without extensively training agents for every task, a simplified method was employed. Instead of using neural networks, we utilized ”dummy” agents. These agents executed predefined actions at set dates during the crop-growth cycle.

The initial phase of task selection involved evaluating various Nitrogen (N) application and irrigation policies on different dates. The farmer’s profit was the metric for policy assessment, acting as the reward. Policies were denoted as a tuple (x, y, z), where each element corresponds to an action from the action space detailed in subsection 3.4.2.

Policies were chosen from a predefined set, with x, y, and z indicating actions executed on specific dates. The combined total of x, y, and z was restricted to 4 to prevent excessive fertilization and irrigation. The date range spanned from planting to harvesting, and various date combinations were tested to gauge policy effectiveness.

In detail, a ”dummy” agent was provided with two tuples: one indicating actions and the other specifying the dates for these actions. Supplying this information as input to Cycles to perform a simulation, and by processing the simulation results as CyclesGym would, yielded a performance metric in k$ per hectare, the standard reward unit in this study.

3.8.2 Task Selection Criteria

Tasks were selected based on the diversity of their best-performing policies. Tasks that required different policies were considered more challenging and diverse, making them suitable for evaluating MTRL.
Through this process, the best-performing policies for each task would be saved, as well as the performances of every policy on every task. At the end, the top-3 performing policies for each task, would be evaluated in the rest of the tasks, and only the tasks that had completely different top-3 best performing policies would be saved. To give an example, if task X has policies $(2,2,0),(3,1,0),(4,0,0)$ as top-3 in performance, while task Y had policies $(0,0,4),(0,1,3),(1,0,3)$ as top-3 in performance, we would save these two tasks, as they are solved by completely different set of policies.

3.8.3 Conclusion

To summarize, instead of having a neural network decide the actions, a simplified agent was employed, with preset actions that were to be taken on specific, predefined dates, allowing us to avoid lengthy trainings, and quickly identify whether two tasks are diverse enough to *not* be solved by similar policies.

The task selection process was crucial for evaluating multi-task learning in crop management. By focusing on tasks that require different policies to reach top performance, it was possible to assess the advantages of multi-task learning in handling diverse and challenging tasks. This approach provided a more robust and meaningful assessment of multi-task learning in crop management.

3.9 Reinforcement Learning Framework

To facilitate the Reinforcement Learning model training, Stable Baselines 3 was used as the RL framework. Stable Baselines 3 is a set of high-quality implementations of reinforcement learning algorithms in Python. It provides a consistent and easy-to-use interface for Reinforcement Learning research and development [41].
Why Stable Baselines 3?

- **Compatibility**: Seamlessly integrates with CyclesGym, providing a unified platform for training and evaluating RL agents.
- **Versatility**: Offers a wide range of pre-implemented RL algorithms, enabling experimentation with different approaches to find the optimal solution for crop management.
- **Efficiency**: Built with a focus on performance and ease of use, facilitating rapid prototyping and development.

3.9.1 Experiment configuration

In Stable Baselines 3, the default network architecture for both the policy and value networks is a multi-layer perceptron (MLP) with two hidden layers. Each hidden layer contains 64 neurons by default. The activation function used for the hidden layers is the TanH function [26]. The output layer of the policy network has a size that corresponds to the number of actions in the action space, while the output layer of the value network has a single neuron.
When explicitly mentioned, some experiments were performed using MLP, with three hidden layers, where each layer contained 512 neurons. The training algorithm is PPO [44]

Early Stopping

Early stopping is a regularization technique used during the training of machine learning models to prevent overfitting [19]. It involves terminating the training process before the model starts to memorize rather than generalize from the training data. This is particularly useful to save computational resources and to ensure the model's performance generalizes well to unseen data. The principle behind early stopping is to monitor the model's performance on a validation dataset and halt the training once the performance ceases to improve. In this work, early stopping was employed as a pragmatic approach to prevent overfitting and excessive training time. Specifically, the training was terminated after the performance did not exhibit more than a 5% change in the error rate for a window of 15 epochs. This practice helped in preventing the potential overfitting and ensuring a robust model performance across different tasks while being resource-efficient.

3.10 Multi-Task Reinforcement Learning

In this section, this work's Multi-Task Reinforcement Learning (MTRL) employed techniques will be we discussed. These techniques were selected based on a review of the literature, identifying them as performing optimally in various multi-task scenarios.

Sequential Task Sampling, named in [28] and found in [38] is a technique where the model weights are updated sequentially, using the rollouts of every tasks - one task at a time. This method is grounded on the principle that focusing on one task at a time can potentially lead to better convergence and data efficiency [28].

To update the DRL model's weights, two methods were used:

- Unitary Scalarization [28] where the updated step is towards the opposite direction of the sum of per-task gradients:
$$\nabla\theta_{\mathrm{MT}} = \sum_{i \in T} \nabla\theta_{L_i}$$
.

- PCGrad[59], where each task gradiented is projected onto the normal plane, defined by the total gradients that the task is conflicting with

Chapter 4

Results

4.1 Evaluation Metrics

It is already established by previous research, that an RL agent can outperform traditional strategies in crop management. However, to evaluate general RL agents, the performance comparison should be against the performance of optimal policies, by trained, task-specific RL agents, and not against the performance of traditional approaches. In essence, the benchmark for a general agent's success is how closely it can emulate the performance of an agent optimized for a specific task. As such, the following metric is used in the graphs, unless specified otherwise:

$$Performance\,\% = \frac{p_B^A}{p_B^B} \times 100\,\% \tag{4.1}$$

where p_B^A is the performance, measured in economical profit of k\$ per hectare of land, of an agent trained in location A and evaluated in location B.

4.2 Reward Function Refinement

To gain a deeper understanding of the complexities in generalizing an RL agent, we explored the idea that a single agent trained in one specific location (Location A) could be deployed and operate efficiently in different environments (Locations B, C...). The experiments were conducted by training the agent in Location A and subsequently deploying and evaluating it in the other locations. A series of experiments were conducted with the aim was to:

- Assess RL agent's ability to generalize and perform in new, unseen environments
- Evaluate which tasks provide knowledge that can be transferred to other tasks
- Evaluate which agents are robust enought to perform well across tasks

As mentioned in Section **??** and based on previous literature, the experiments were using Equation 3.3 as Reward Function, which was defined solely on the profit based on crop yield,

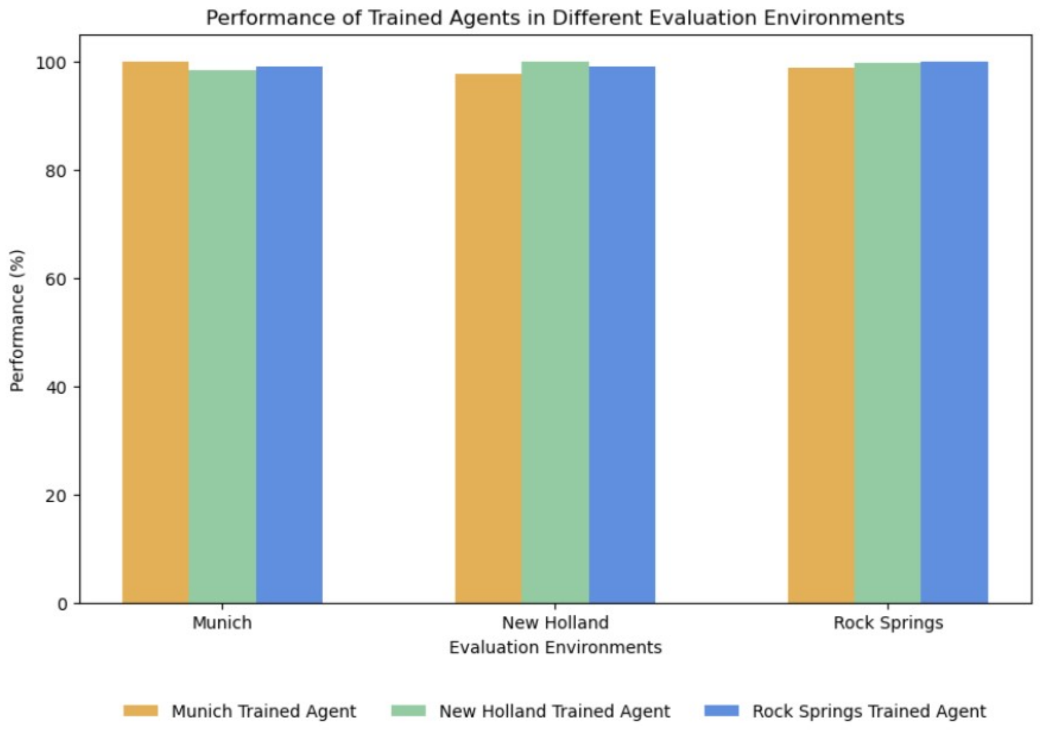

Figure 4.1: Performance of agents in Munich, New Holland, Rock Springs and evaluated in all of them as well. Metric is Equation 4.1

subtracting the cost of fertilization.

A significant drawback that was observed during the experiments was the excessive use of nitrogen. The reward function was primarily focused on maximizing economic benefits, i.e., the profit from yield minus the cost of nitrogen. This led to an imbalance where the agent was not sufficiently penalized for over-utilizing nitrogen, resulting in an inefficient strategy.

As shown in Figure 4.1, the initial training environment is almost irrelevant, as an agent trained with data from Munich, Germany will perform almost as well in New Holland, USA, as an agent trained for that specific location. This performance is counterintuitive, as one would expect that an agent trained in Europe would not perform perform nearly as well as an agent trained for a specific location in the US. To further investigate this, experiments were run on the diverse tasks that were identified from the created dataset, as mentioned in Section 3.8. However, the counterintuitive results were, again, repeated, as shown in Figure 4.2.

After a consultation meeting with agricultural experts, it was apparent that the reward function was overly simplistic and did not adequately represent the multifaceted real-world costs involved in crop management. The experts highlighted the need to incorporate additional costs such as leaching fees, irrigation fees, and balance fees [32], in order to create a more realistic

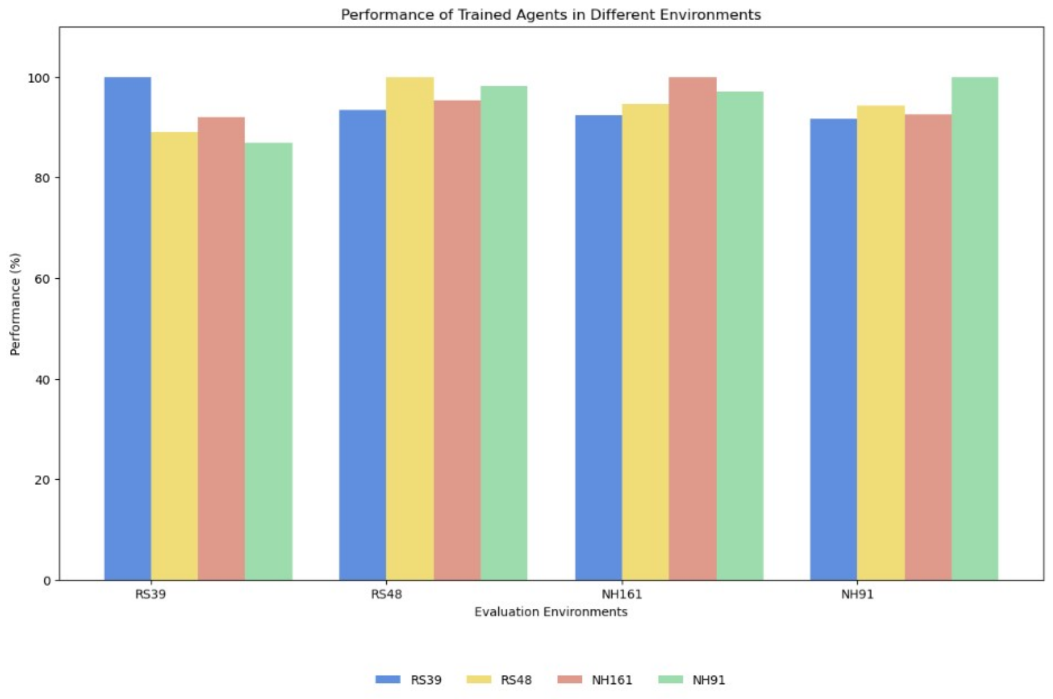

Figure 4.2: Performance of agents in RS39, RS49, NH161, NH91 and evaluated in all of them as well. These tasks are from the generated dataset, and selected to be diverse and dissimilar, as mentioned in Section 3.8. It is notable that even in these diverse tasks, the worst performance of an agent is 85%. Further information on these tasks can be found in Metric is Equation 4.1

reward function that reflects the economic trade-offs that farmers face. As a result, the reward function was updated to incorporate these factors.

The reward function, as expressed in Equation 3.4, takes into account not only the profit from crop yield and the cost of fertilizer, but the cost of irrigation, as well as multiple possible environmental-impact fees. As shown in Figure 4.3, the performance of the agents trained in diverse tasks differ significantly, as one would expect.

With this modification in the reward function, the evaluation of a developed MTRL agent is substantially more robust. Previously, using the initial reward function, all agents appeared to perform almost uniformly across tasks, which would make it challenging to understand if a developed MTRL agent could generalize or not. Now, given the clear performance differences among trained agents across tasks, if a developed MTRL agent excels in multiple tasks, it is a clear indication of its generalization capabilities.

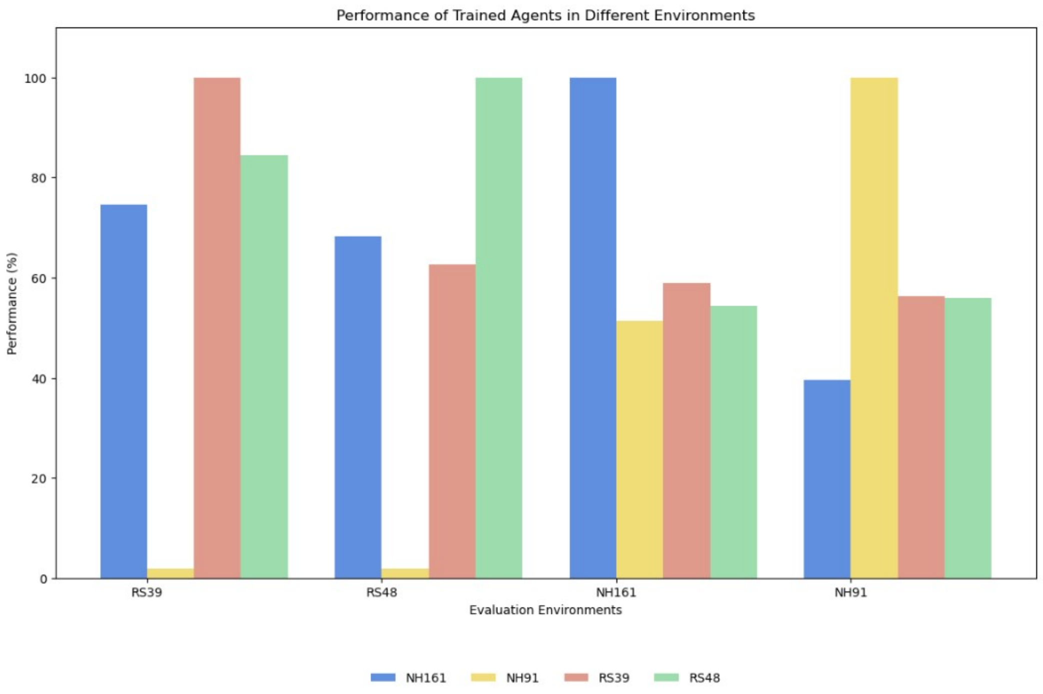

Figure 4.3: Performance of agents in RS39, RS49, NH161, NH91 and evaluated in all of them as well. These tasks are from the generated dataset, and selected to be diverse and dissimilar, as mentioned in Section 3.8. It is notable that even in these diverse tasks, the worst performance of an agent is 85%. Further information on these tasks can be found in Metric is Equation 4.1

4.3 Multi-Task Reinforcement Learning Experiments

In the pursuit of understanding the generalization capabilities of a single agent in reinforcement learning (RL) applied to crop management, a series of progressive ideas were explored. Initially, the concept of generalization was investigated, whereby a single agent trained in one location (A) was evaluated across diverse tasks, as shown in Figures 4.3 and 1.4. However, as these tasks were diverse, the subset of tasks that were dissimilar only across temperature were used to evaluate whether performance deteriorates monotonically in progressively dissimilar tasks. The focus then shifted to the efficiency, generalization, and speed of multi-task learning models, including the limits of concurrent tasks that can be learned, the largest trainable area for specific and intermediate tasks, and addressing potential conflicts in gradients using optimizers, such as PCGrad.

In the following experiments, tasks from the dataset, as explained in Section 3.7, are used. To clarify, as **task**, we define the set of input files used for a Cycles simulation, and specifically, the set of weather, soil and crop files.

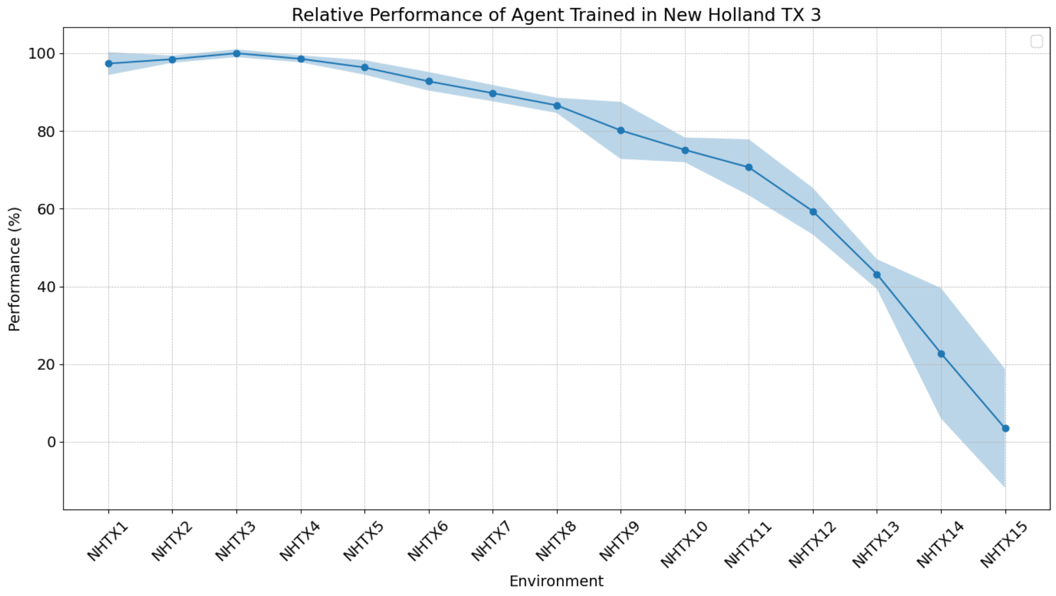

Figure 4.4: Performance of an NHTX3-trained agent in each of the NTHX tasks

4.3.1 Selected Diverse Tasks

As mentioned in Section 3.8, multiple diverse tasks were identified in the dataset. However, the initial evaluation of hardcoded policies only served as an indicator on possible tasks. To solidify the selection of such tasks, an RL agent was trained for each, and evaluated across all the candidates. Four were selected as the most diverse, and are most commonly used in the following experiments.

- Rock Springs 39 (RS39)
- Rock Springs 48 (RS48)
- New Holland 91 (NH91)
- New Holland 161 (NH161)

Their performance is shown in Figure 4.3. For more information about the meaning of each number, see Section 3.8.

4.3.2 Monotonic Plot

In the monotonic plot experiment, the performance of a single agent is evaluated across progressively dissimilar tasks. Specifically, we consider tasks NHTX1 through NHTX15, where the maximum daily temperature is incrementally increased. The rationale behind this is to understand how an agent's performance might decrease in increasingly divergent conditions. In Figure 4.4 the performance of an agent trained in NHTX3 and evaluated in tasks NHTX1,...,NHTX15 is decreasign monotonically, confirming the hypothesis that increasing the difference between tasks, the performance of an agent is decreased.

4.3.3 Multi-Task Reinforcement Learning evaluation in similar tasks

Building on the concept of Multi-Task Reinforcement Learning, this experiment focuses on sequential sampling MTRL training, as detailed in 2.2.2. The agent is trained on four specific tasks: RSTX1, RSTX5, RSTX10, and RSTX15, which again differ only on the daily maximum temperature, and evaluated across RSTX1,...,RSTX15. As shown in Figure 4.5, the agent not only performs well on the training tasks but also interpolates effectively on the intermediate evaluation tasks. This showcases the agent's ability to generalize, leading to a speed-up of 50-75% compared to training a separate agent for each task, when the total training steps are taken into account.

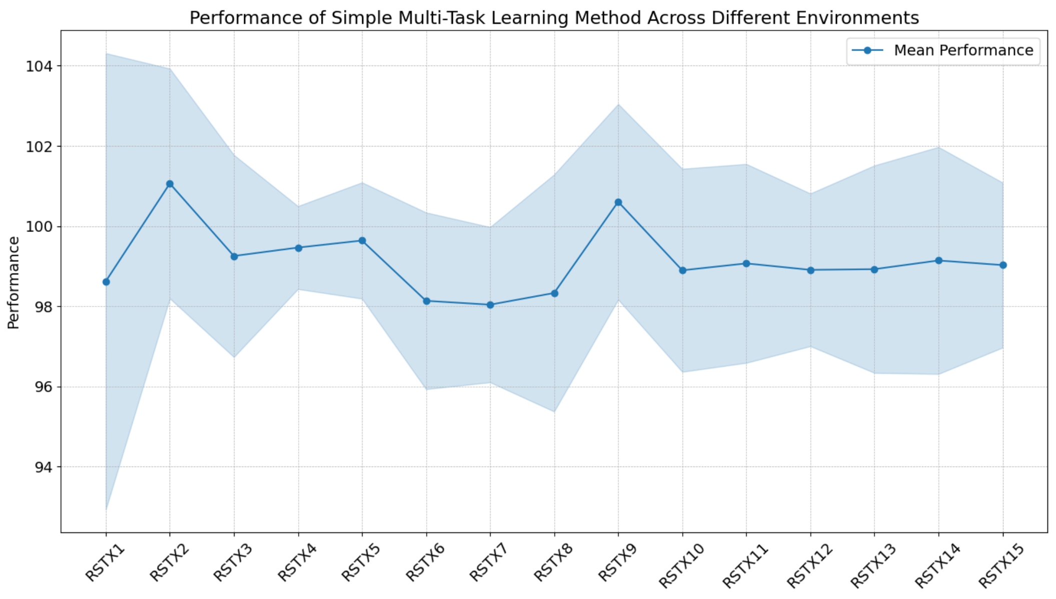

Figure 4.5: Performance of an MTRL agent, trained in RSTX1, RSTX5, RSTX10, and RSTX15 and evaluated in each of the RSTXx tasks

4.3.4 Multi-Task Reinforcement Learning evaluation in diverse tasks

While the previous experiment demonstrated MTRL's potential in similar tasks, as the only difference was maximum daily temperature, this experiment pushes the boundaries by testing the agent on four dissimilar tasks: RS39, RS48, NH91, and NH161, as identified in Section 4.3.1. As shown in Figure 4.6, the results are promising: MTRL successfully solves all the tasks, achieving a 50% speed-up compared to the total individual agent training steps.

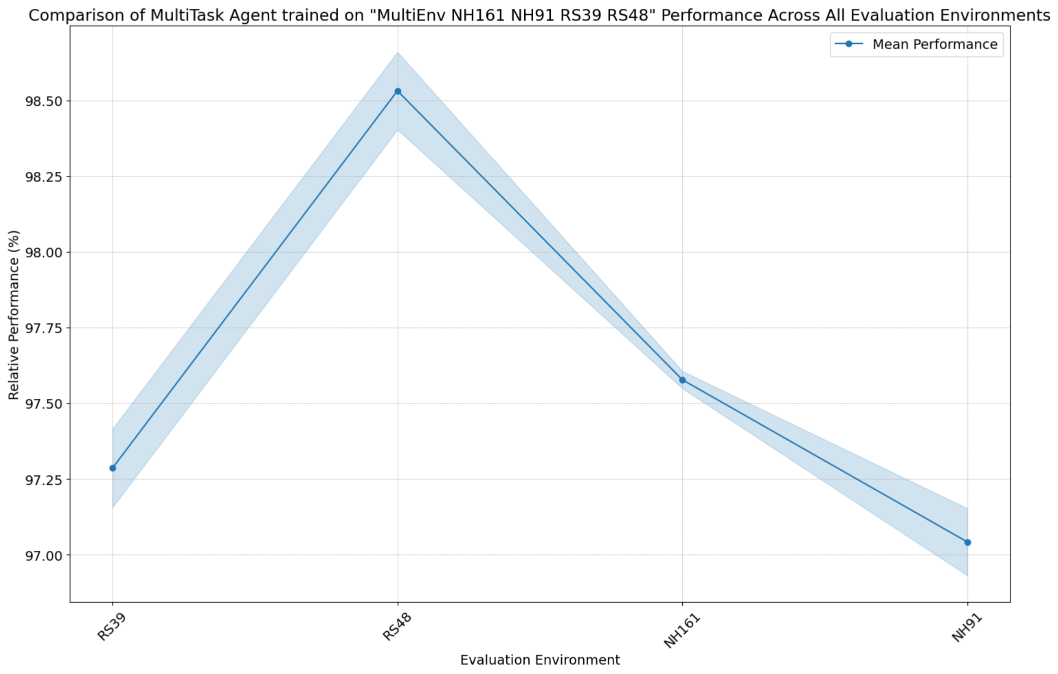

Figure 4.6: Performance of an RL agent trained on {RS39, RS48, NH91, NH161}, evaluated across all four tasks.

4.3.5 Multi-Task Reinforcement Learning with PCGrad in similar tasks

PCGrad, known for addressing conflicting gradients, is introduced in this experiment. The agent is trained on tasks RSTX1, RSTX5, RSTX10, and RSTX15. However, when evaluated across RSTX1-15, an MTRL model without PCGrad outperforms its counterpart with PCGrad, as shown in Figure 4.7.

4.3.6 Multi-Task Reinforcement Learning with PCGrad in the diverse Tasks

Given the results of the previous experiment, we hypothesized that PCGrad's effectiveness might be reduced when the algorithm is applied to tasks that are similar, as PCGrad's main focus is conflicting gradients between tasks. To test this, we repeated the experiment with the four diverse tasks mentioned earlier. As shown in Figure 4.8, the results confirmed the hypothesis: PCGrad performed equally or better in three out of the four tasks. In Figure 4.9, we show again the effectiveness of PCGrad, compared to non-PCGrad training, in diverse locations. The locations used were the same as in 4.2, where we showed that single agent training is failing even in locations that are close distance-wise. This underscores the idea that MTRL can generalize compared to a single agent, and that PCGrad is better suited for diverse tasks and aligns with this research's goal, to develop general agents that manage *diverse* fields.

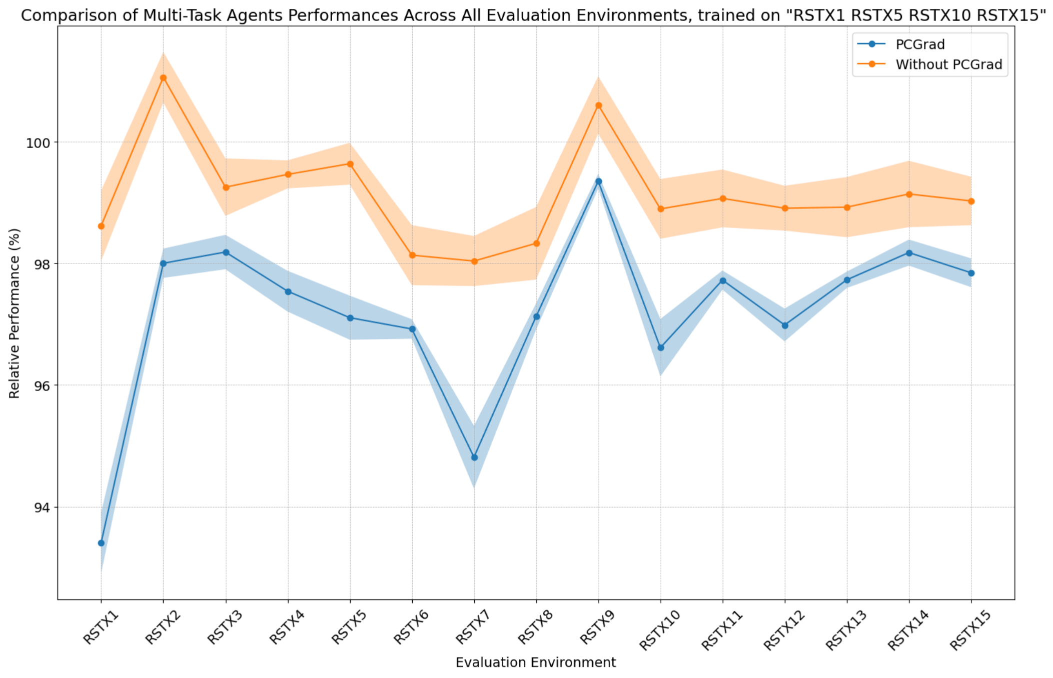

Figure 4.7: Performance of two RL agents trained on {RSTX1, RSTX5, RSTX10, RSTX15}, evaluated across all 15 RSTXx tasks. One was trained with the PCGrad optimizer, and one without.

4.3.7 Multi-Task Reinforcement Learning with PCGrad in combining diverse and similar tasks

To further validate our findings regarding PC, we scaled up the experiment. The agent was trained on three RSTX tasks(RSTX1, RSTX10, RSTX15) and the four diverse tasks (RS39, RS48, NH91, NH161). When evaluated, it was evident that the use of PCGrad significantly enhanced the agent's ability to generalize across all tasks, as shown in Figure 4.10, confirming that PCGrad is a significant improvement method for MTRL.

4.3.8 Multi-Task Reinforcement Learning with Large Networks (with/without PCGrad)

In this experiment, we explored the impact of network size on performance. While the ”Default Network” comprised a shared parameter network of 2 MLPs with 64 neurons per layer, we introduced a ”Large Network” comprising 3 MLPs with 512 neurons per layer. The results, in Figure 4.11 show that without PCGrad, the large network's performance dropped. However, when combined with PCGrad, the ”Large Network”'s performance was almost on par with the performance of the ”Default Network” using PCGrad. This experiment supports our previous hypothesis on PCGrad's potential in diverse tasks. However, the benefits of using a larger network remain unclear and should be investigated further.

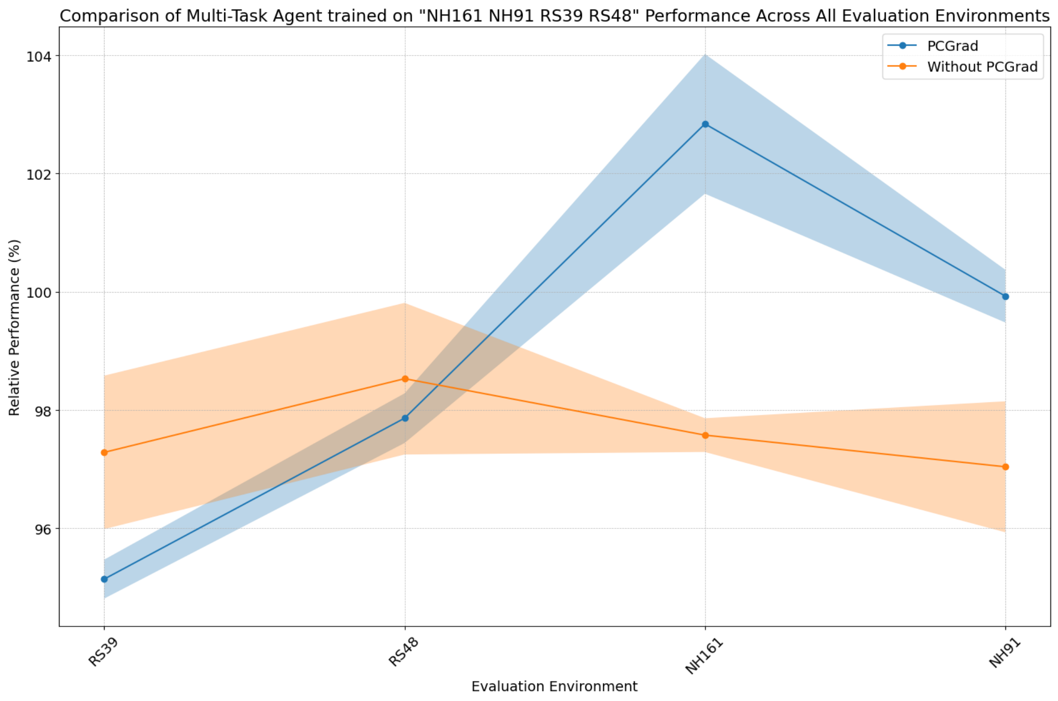

Figure 4.8: Performance of two RL agents trained on {RS39, RS48, NH91, NH161}, evaluated across all four tasks. One was trained with the PCGrad optimizer, and one without.

4.3.9 Minimum number of required tasks

As the number of training tasks in Multi-Task Reinforcement Learning affects performance and computational resources needed, it is crucial to identify the required tasks for optimal performance, and their number. In assessing the agents trained with and without the PCGrad optimizer, a series of experiments were conducted. These were designed to determine the minimal set of training tasks required for the agents to achieve optimal performance across different tasks. Three different task sets were considered, based on temperature variations, to assess the agents' generalization capabilities.

As expected, based on Figure 4.7, across all experiments that were performed on the RockSpringsTX set of tasks, the agent that was trained without PCGrad showed superior performance.

Figure 4.12 shows that there is a noticeable decrement in performance for the best performing agent on intermediary tasks. This suggests that training on temperature extremes alone does not equip the agent to generalize well across the entire range.

Figure 4.13 further investigates this by training the agents on two intermediate temperature tasks, RSTX7 and RSTX12. While the agent not using PCGrad still outperforms its counterpart, it shows, again, a decline in performance on the lower temperature tasks.

To explore a more balanced training set, Figure 4.14 shows the performance of the agents on three tasks: RSTX1, RSTX7, and RSTX15. With this mix of extreme and intermediate tasks, the best-performing agent performs extremely well across all the evaluation environments.

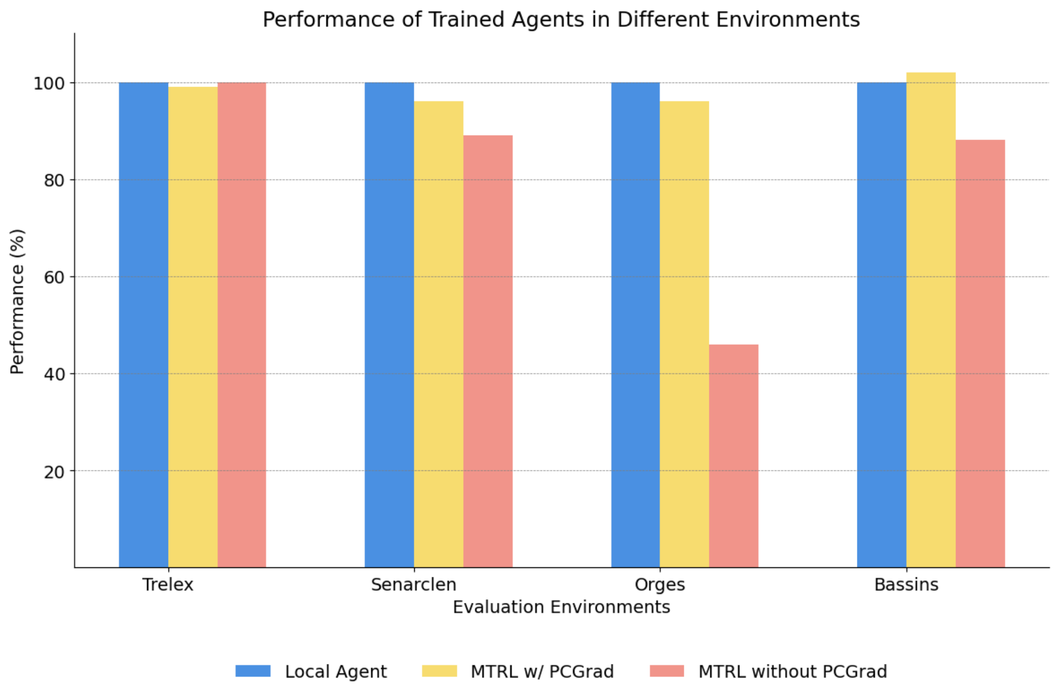

Figure 4.9: Performance comparison of RL agents trained on {Trelex, Senarclen, ORges, Bassins}. The local agent is an RL agent trained on a single environment and its performance is the optimal, while the other two were trained with Multi-Task Reinforcement Learning with and without PCGrad respectively. The use of PCGrad leads to superior generalization across the locations.

This supports the importance of the training task selection. However, it is unclear which tasks should be selected in a more realistic scenario, and by what criteria.

4.3.10 Realistic experiments

To conclude the experimentation with MTRL techniques and PCGrad optimizers, we run an experiment in a more realistic scenario, by retrieving data on 8 swiss locations, namely:

- Zurich
- Bern
- Basel
- Geneva
- Senarclens
- Ogres

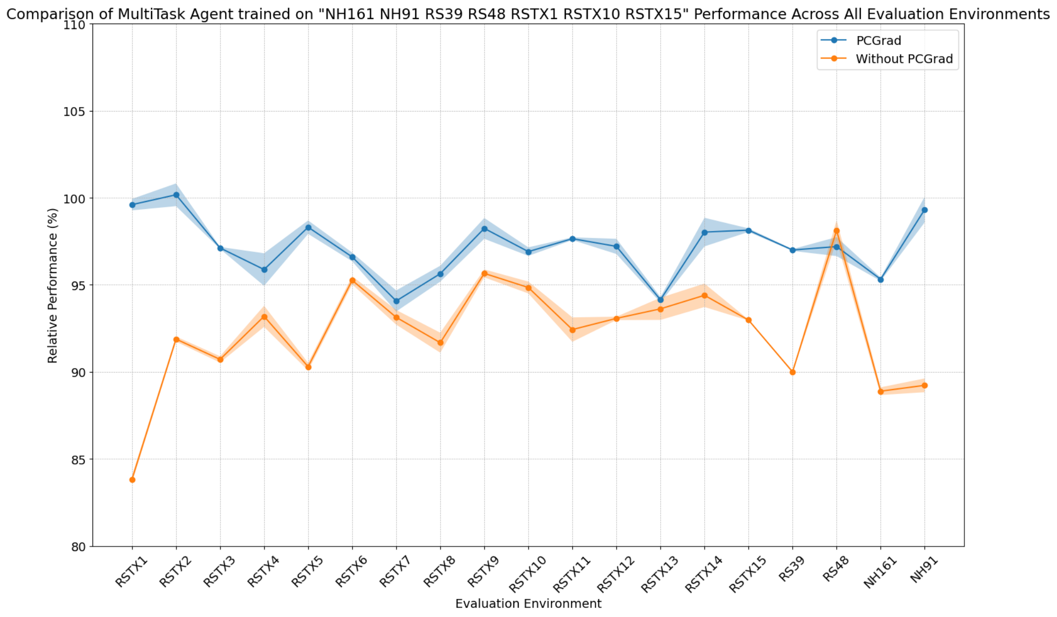

Figure 4.10: Performance of two RL agents trained on {RS39, RS48, NH91, NH161, RSTX1, RSTX10, RSTX15}, evaluated across all 15 RSTXx tasks and the 4 diverse tasks. One was trained with the PCGrad optimizer, and one without.

- Trelex
- Bassins

The purpose of the experiment is to examine the out-of-distribution performance of our agent, in the weather setting of the same country, without using synthetic tasks. Towards that goal, we trained two agent with and without using PCGrad, on the first 4 of these tasks, and evaluated on all of them, as showed in 4.15. The results show that both agents were capable of demonstrating a level of generalization to the out-of-distribution tasks. This suggests that training on a subset of representative tasks within the same geographical region (in this case, Switzerland) can equip agents to handle novel tasks within that region. However, it's worth noting that certain drops in performance were observed for both agents across these out-of-distribution tasks. This suggests that while there is a level of inherent generalization, there still exist nuances or specificities in the unseen tasks that challenge the agents' performance.

4.3.11 Conclusion

In summary, these experiments provide valuable insights into the capabilities and limitations of Multi-Task Reinforcement Learning, PCGrad, and network sizes. The consistent theme is the ability for MTRL agnets to generalize across multiple tasks and the potential of PCGrad to further optimize such agents, especially in diverse tasks.

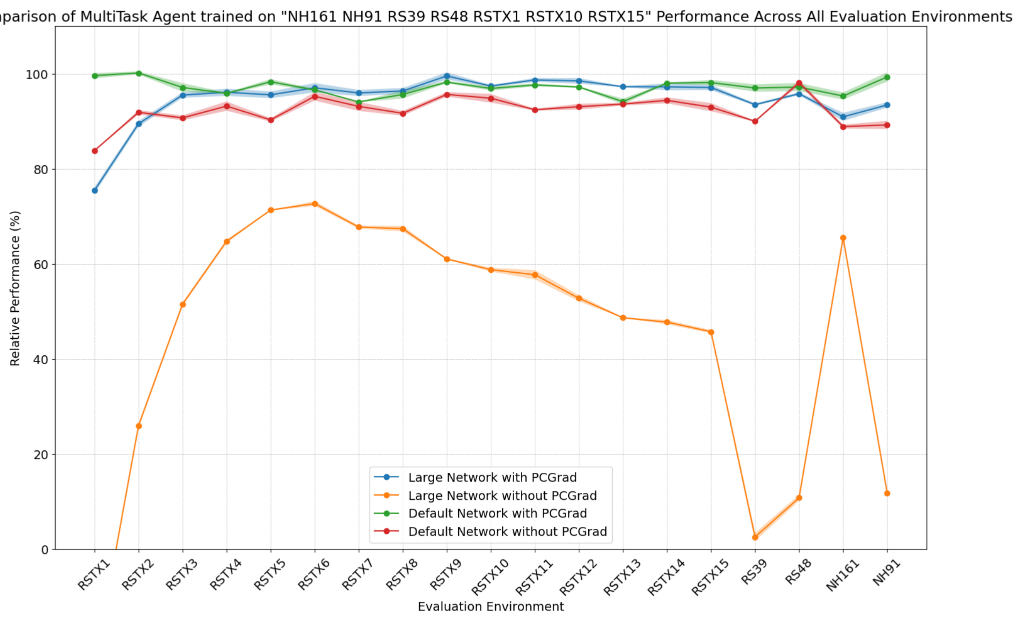

Figure 4.11: Comparing 4 agents of each combination of {with PCGrad, without PCGrad} and {Large Network, Default Network}. Large Network with PCGrad shows great improvement in performance compared to Large Network without PCGrad. However, both fall short compared to the performance of Default Network.

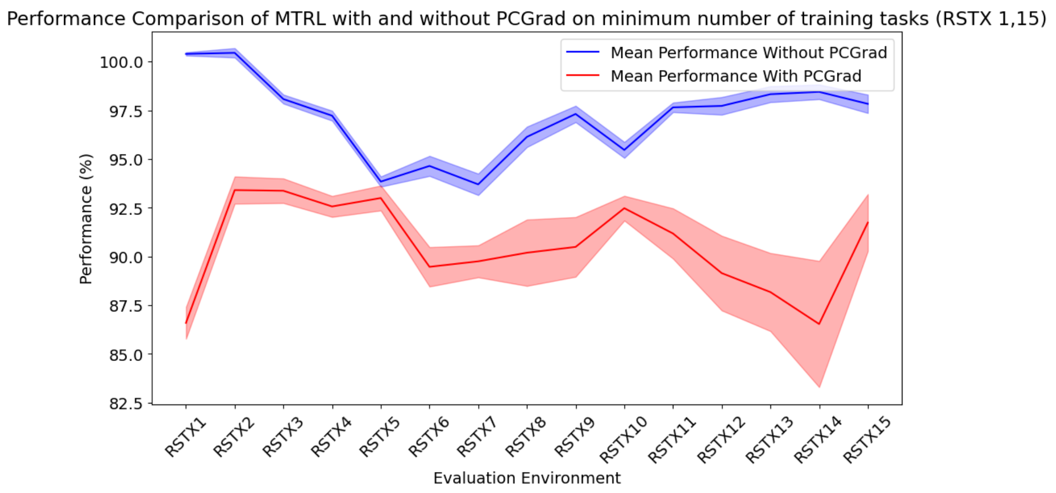

Figure 4.12: Comparing performances of two agents (with and without PCGrad optimizer during training), trained on the RSTX1 and RSTX15 tasks, the performance of the non-PCGrad using agent stands out. In an effort to minimize the training tasks used, we chose the two most extreme ones - lowest and highest temperature. A significant performance drop is observed in the middle tasks, by the best performing agent.

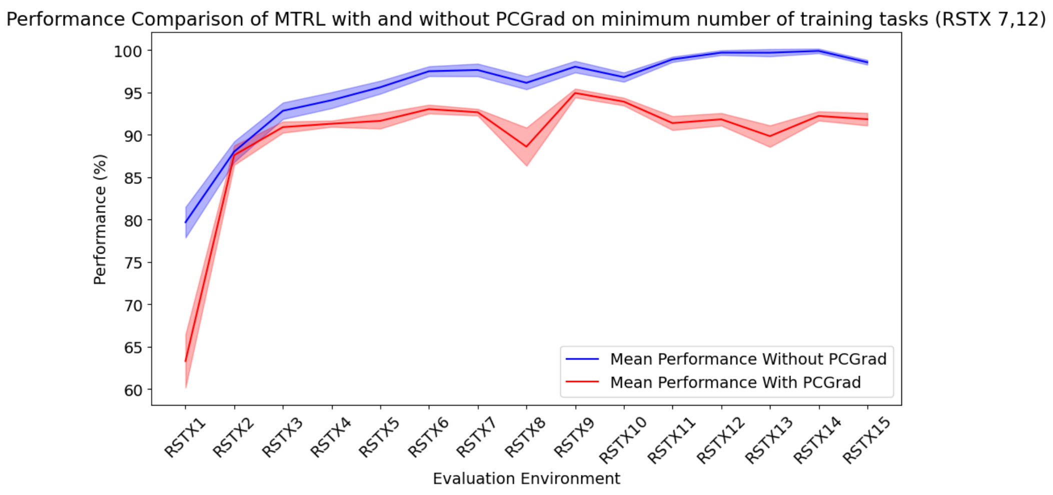

Figure 4.13: Comparing performances of two agents (with and without PCGrad optimizer during training), trained on the RSTX7 and RSTX12 tasks, the performance of the non-PCGrad using agent stands out. In an effort to minimize the number of training tasks used, we chose two middle temperature tasks. However, a significant performance drop is observed in the low temperature tasks.

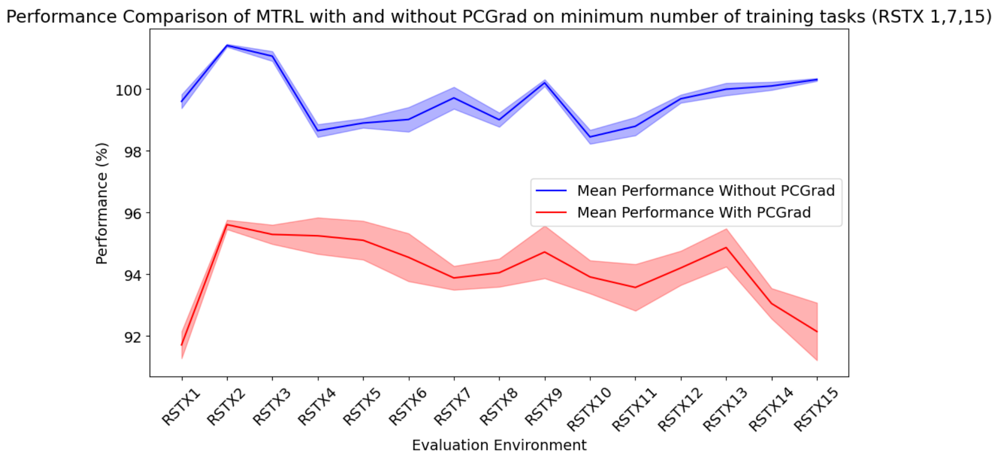

Figure 4.14: Comparing performances of two agents (with and without PCGrad optimizer during training), trained on the RSTX1, RSTX7 and RSTX15 tasks, the performance of the non-PCGrad using agent stands out. In an effort to minimize the training tasks used, we chose three out of 15 - two in the extremities and one in the middle temperature range. The best performing agent performs well across the board.

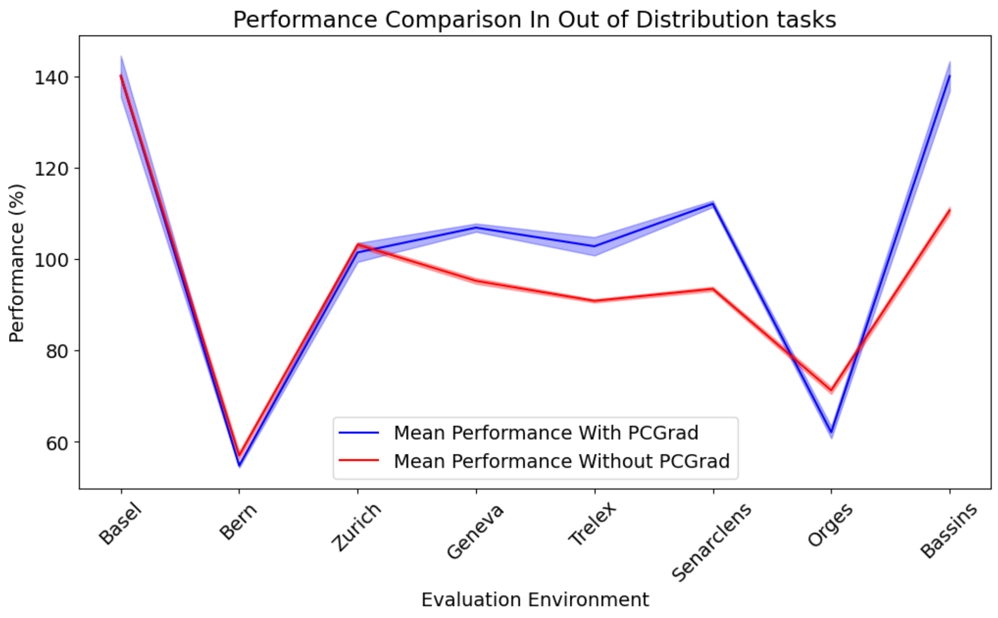

Figure 4.15: Comparing performances of two agents (with and without PCGrad optimizer during training), on 8 swiss locations. The agents were trained on Zurich, Bern, Basel, Geneva and evaluated on the training tasks and on Senarclens, Ogres, Trelex, Bassins. We observe a generalization in tasks that are out of distribution. However, drops in performance exist.

Chapter 5

Conclusion & Discussion

5.1 Conclusions

This thesis delved deep into crop management using reinforcement learning (RL), specifically focusing on the generalization capabilities of RL agents. Through a set of experiments, we investigated the potential and limitations of various Multi-Task Reinforcement Learning training and optimization techniques. To conclude, we summarize the key findings of this thesis, and discuss the limitations and future interesting work.

5.1.1 Key Findings

Our research confirm the claims of prior studies regarding RL agents outperforming more traditional strategies in crop management. The first step was redefining common agricultural reward functions by including additional real-world costs, such as environmental-impact fees and irrigation costs, which resulted in a more robust evaluation of agents, and more realistic behavior of the trained agents.

Then, we showed the importance of generalization in this specific application. Training a single agent for each geographical location is simply impossible in terms of time and required comutational resources. To tackle this challenge, we investigated the use of Multi-Task Reinforcement Learning (MTRL). Our experiments showcased the power of MTRL in enabling agents to generalize across tasks. Whether the tasks were similar in nature or diverse, MTRL-equipped agents demonstrated a significant speed-up in training and an ability to interpolate between the tasks they were trained on.

Furthermore, we showed PCGrad's potential in environments with diverse tasks. While in similar tasks it did not provide a discernible advantage, the improvement was evident when agents were trained across diverse tasks, making it a promising technique for real-world scenarios where diversity is the norm rather than the exception.

The exploration into different network sizes indicated that larger networks, although computationally more intensive, did not always guarantee better performance. However, when combined with optimization techniques like PCGrad, larger networks demonstrated competitive performance, suggesting that the further investigation between network architecture and optimization methods is a promising future direction.

Our results highlighted the importance of training task selection in MTRL. A ”suitable” task identification method could offer better generalization performance, specifically in diverse scenarios.

Finally, in a realistic setting involving Swiss locations, our trained agents showcased generalization to out-of-distribution tasks, though there were observable performance drops. This reiterates the need for comprehensive training data that covers potential edge cases, alongside task identification techniques.

5.2 Limitations and Future Research

This study explores the application of Multi-Task Reinforcement Learning (MTRL) in agriculture, building upon the CyclesGym framework. However, it contains certain limitations that are discussed in the following subsections. These limitations include issues related to data acquisition, task similarity analysis, model architecture, the selected Reinforcement Learning (RL) and MTRL algorithms, and the simulation models used. Acknowledging these limitations provides a clearer understanding of the study’s findings and outlines areas for future research.

5.2.1 Data Limitations

As the data used in this study were collected from online sources, this approach has limitations such as potential inaccuracies, inconsistencies, or missing values in the data which could affect the model’s performance. Additionally, the data may not capture all relevant variables for agricultural field management, possibly leading to an oversimplified model of real-world scenarios.

5.2.2 Task Similarity

The MTRL agent showed promising performance across multiple tasks. However, the similarity or dissimilarity among these tasks is not clear. Understanding task similarity could help in deploying trained agents across similar tasks globally, improving performance and reducing the need for retraining. A systematic approach to measure task similarity and its impact on MTRL performance is needed.

5.2.3 Model Architecture

A few model architectures were explored in this study, although it was not the primary focus. Other architectures might perform better or worse. The chosen architectures may have biases or assumptions affecting the findings’ generalizability. Evaluating different architectures could be a focus for future research to find the optimal model structure for this domain.

5.2.4 RL Algorithm

The RL algorithm choice can significantly impact model performance. This study investigated a limited number of RL algorithms, leaving many other potentially effective algorithms unexplored. Different RL algorithms may have varying efficiency, convergence speed, and robustness in handling multi-task learning in this scenario. Exploring alternative RL algorithms could be beneficial.

5.2.5 MTRL Algorithm

The MTRL algorithm was selected based on literature reviews. However, there are other multi-task optimizers and methods that might offer different performance characteristics. The approach to context representation in the MTRL algorithm could also impact performance across tasks. Exploring various MTRL algorithms, multi-task optimizers, and context representation strategies could provide more insights into achieving higher performance in multi-task reinforcement learning scenarios.

5.2.6 Sim2Real Gap

The simulation-to-reality (Sim2Real) [22] gap represents a significant challenge in transitioning the learned policies of the MTRL agent to real-world agricultural management tasks. The performance observed within the simulated environment may not accurately reflect the performance in real-world scenarios due to various factors such as unmodeled dynamics, real-world uncertainties, and discrepancies between the simulated and real-world data. The Sim2Real gap underscores the necessity for rigorous real-world validation to ascertain the efficacy and applicability of the developed MTRL agent in actual agricultural settings.

5.2.7 Cycles Model

The simulation environment utilized in this research employs the Cycles mechanistic model to mimic agricultural dynamics. Nonetheless, the accuracy of the Cycles model in capturing the real world dynamics is a crucial factor; any inconsistencies or inaccuracies in the model could potentially mislead the training of the RL agent. The effect of the research output is dependent on the accuracy and reliability of the Cycles model. More precise mechanistic simulation models might yield different training outcomes and subsequently affect the RL agent's performance.

5.2.8 Multi-Year Effects

The training conducted in this study spanned one-year simulations. However, agriculture is inherently a multi-year endeavor with complex interactions over time, encompassing crop rotations, weather variations, soil fertility dynamics, and other long-term factors. An RL agent trained and validated on single-year simulations may exhibit deteriorated performance in multi-year scenarios. The discount factor in reinforcement learning, which determines the agent's consideration for future rewards, plays a crucial role in long-term decision-making. The simplistic single-year training might overlook the intricate and cumulative effects that evolve over multiple years, necessitating further research to explore multi-year training and evaluation to ensure the robustness and longevity of the agent's performance.

5.2.9 Future Work Ideas

1. **Incorporation of Remote Sensing Data:** Satellite and drone imagery offer high-resolution insights into environmental factors, which are crucial for precision agriculture. The aim is to leverage these data sources to enhance the models' understanding of on-ground conditions. An experiment to validate this approach would involve training models using remote sensing data combined with traditional parameters. The subsequent evaluation would compare the models' ability to predict optimal farming strategies, potentially leading to more informed agricultural decisions.

2. **Real-time Adaptation to Weather Changes:** Weather plays a pivotal role in agricultural outcomes. The proposition is to have models that can swiftly adapt their strategies based on real-time and forecasted weather conditions, ensuring that farming practices remain optimal even under unpredictable circumstances. To assess the efficacy of this real-time adaptability, one could simulate sudden weather changes and evaluate how well the models adjust their recommendations.

3. **Human-in-the-Loop Learning:** While computational models are powerful, human expertise, especially in a field like agriculture, remains invaluable. The objective is to integrate farmer feedback into the learning process, creating a synergy between human knowledge and algorithmic insights. An experiment in this direction would involve collecting and implementing human feedback during model training. The assessment would then gauge improvements in the model's practical applicability and overall performance.

4. **Ethical and Sustainable Farming Practices:** With growing concerns about the environment and ethical farming, it's essential for models to factor in these considerations. The intent is to ensure that the model's reward function aligns with sustainable and ethical farming standards. To validate this approach, models trained with varying ethical considerations could be compared. The evaluation would focus on both economic performance and key sustainability metrics, ensuring a balance between profitability and ethical considerations.

Bibliography

[1] Cycles github repository. https://github.com/PSUmodeling/Cycles/releases/tag/v1.2.1. Accessed: 2023-09-10.

[2] Department of economic united nations and social affairs. world population prospects., 2022. URL https://population.un.org/wpp/.

[3] Mukhtar Ahmed, Shakeel Ahmad, Muhammad Ali Raza, Uttam Kumar, Muhammad Ansar, Ghulam Abbas Shah, David Parsons, Gerrit Hoogenboom, Taru Palosuo, and Sabine Seidel. Models calibration and evaluation. *Systems modeling*, pages 151–178, 2020.

[4] Oguzhan Alagoz, Heather Hsu, Andrew J Schaefer, and Mark S Roberts. Markov decision processes: a tool for sequential decision making under uncertainty. *Medical Decision Making*, 30(4):474–483, 2010.

[5] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6): 26–38, 2017.

[6] Awasthi Lalit Attri, Ishana and Teek Sharm. Machine learning in agriculture: a review of crop management. *Springer*, 2023. URL https://link.springer.com/article/10.1007/s11042-023-16105-2.

[7] Lefteris Benos, Aristotelis C. Tagarakis, Georgios Dolias, Remigio Berruto, Dimitrios Kateris, and Dionysis Bochtis. Machine learning in agriculture: A comprehensive updated review. *Sensors*, 21(11), 2021. ISSN 1424-8220. doi: 10.3390/s21113758. URL https://www.mdpi.com/1424-8220/21/11/3758.

[8] Schubert J. De Wit A. Lazebnik J. Hutjes R. Van der Grijn G. Boogaard, H. Agrometeorological indicators from 1979 to present derived from reanalysis. copernicus climate change service (c3s) climate data store (cds). 2020. doi: 10.24381/cds.6c68c9bb.

[9] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *CoRR*, abs/1606.01540, 2016. URL http://arxiv.org/abs/1606.01540.

[10] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997. URL https://link.springer.com/article/10.1023/A:1007379606734.

[11] Mengting Chen, Yuanlai Cui, Xiaonan Wang, Hengwang Xie, Fangping Liu, Tongyuan Luo, Shizong Zheng, and Yufeng Luo. A reinforcement learning approach to irrigation decision-making for rice using weather forecasts. *Agricultural Water Management*, 250: 106838, 2021. URL https://www.sciencedirect.com/science/article/abs/pii/S037837742100058X.

[12] Guangran Cheng, Lu Dong, Wenzhe Cai, and Changyin Sun. Multi-task reinforcement learning with attention-based mixture of experts. *IEEE Robotics and Automation Letters*, 8(6):3812–3819, 2023. doi: 10.1109/LRA.2023.3271445.

[13] Michael Crawshaw. Multi-Task Learning with Deep Neural Networks: A Survey, September 2020.

[14] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, and Koray Kavukcuoglu. IMPALA: Scalable distributed deep-RL with importance weighted actor-learner architectures. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1407–1416. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/espeholt18a.html.

[15] Romain Gautron, Odalric-Ambrym Maillard, Philippe Preux, Marc Corbeels, and Régis Sabbadin. Reinforcement learning for crop management support: Review, prospects and challenges. *Computers and Electronics in Agriculture*, 200:107182, 2022.

[16] Romain Gautron, Odalric-Ambrym Maillard, Philippe Preux, Marc Corbeels, and Régis Sabbadin. Reinforcement learning for crop management support: Review, prospects and challenges. *Computers and Electronics in Agriculture*, 200:107182, 2022. ISSN 0168-1699. doi: https://doi.org/10.1016/j.compag.2022.107182. URL https://www.sciencedirect.com/science/article/pii/S0168169922004999.

[17] Romain Gautron, Emilio J Padrón, Philippe Preux, Julien Bigot, Odalric-Ambrym Maillard, and David Emukpere. *gym-DSSAT: a crop model turned into a Reinforcement Learning environment*. PhD thesis, Inria Lille, 2022.

[18] Vincent Gitz, Alexandre Meybeck, L Lipper, C De Young, and S Braatz. Climate change and food security: risks and responses. *Food and Agriculture Organization of the United Nations (FAO) Report*, 110:2–4, 2016.

[19] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. URL http://www.deeplearningbook.org/.

[20] Jeroen C. J. Groot, Walter A. H. Rossing, and Andre Jellema. Multi-objective optimization and design of farming systems. *Agricultural Systems*, 110:63–77, 2012. URL https://www.sciencedirect.com/science/article/abs/pii/S0308521X15300589.

[21] Matteo Hessel, Hubert Soyer, Lasse Espeholt, Wojciech Czarnecki, Simon Schmitt, and Hado van Hasselt. Multi-task deep reinforcement learning with popart, 2018.

[22] Sebastian Höfer, Kostas Bekris, Ankur Handa, Juan Camilo Gamboa, Melissa Mozifian, Florian Golemo, Chris Atkeson, Dieter Fox, Ken Goldberg, John Leonard, et al. Sim2real in robotics and automation: Applications and challenges. *IEEE transactions on automation science and engineering*, 18(2):398–400, 2021.

[23] Wenlong Huang, Igor Mordatch, and Deepak Pathak. One policy to control them all: Shared modular policies for agent-agnostic control. In *International Conference on Machine Learning*, pages 4455–4464. PMLR, 2020.

[24] Leslie Pack Kaelbling, Michael L. Littman, and Andrew W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

[25] Melanie Kah, Nathalie Tufenkji, and Jason C White. Nano-enabled strategies to enhance crop nutrition and protection. *Nature nanotechnology*, 14(6):532–540, 2019.

[26] Bekir Karlik and A Vehbi Olgac. Performance analysis of various activation functions in generalized mlp architectures of neural networks. *International Journal of Artificial Intelligence and Expert Systems*, 1(4):111–122, 2011.

[27] Armen R Kemanian, Yuning Shi, Charles M White, Felipe Montes, Claudio O Stöckle, David R Huggins, Maria Laura Cangiano, Giovani Stefani-Faé, and Rachel K Nydegger Rozum. The cycles agroecosystem model: Fundamentals, testing, and applications. *Testing, and Applications*, 2022.

[28] Vitaly Kurin. *A minimalist approach to deep multi-task learning*. PhD thesis, University of Oxford, 2022.

[29] Seanie Lee, Hae Beom Lee, Juho Lee, and Sung Ju Hwang. Sequential reptile: Inter-task gradient alignment for multilingual learning. *arXiv preprint arXiv:2110.02600*, 2021.

[30] Konstantinos G. Liakos, Patrizia Busato, Dimitrios Moshou, Simon Pearson, and Dionysis Bochtis. Machine learning in agriculture: A review. *Computers and Electronics in Agriculture*, 151:556–566, 2018. URL `https://www.sciencedirect.com/science/article/abs/pii/S0168169918301525`.

[31] Odalric-Ambrym Maillard. Farm-gym: A modular reinforcement learning platform for agronomy games. 2022.

[32] German Mandrini, Cameron Mark Pittelkow, Sotirios Archontoulis, David Kanter, and Nicolas F. Martin. Exploring Trade-Offs Between Profit, Yield, and the Environmental Footprint of Potential Nitrogen Fertilizer Regulations in the US Midwest. *Frontiers in Plant Science*, 13, 2022. ISSN 1664-462X.

[33] Javier Mateo-Sagasta, S Marjani Zadeh, and Hugh Turral. More people, more food, worse water? a global review of water pollution from agriculture. *Food and Agriculture Organization of the United Nations (FAO) Report*, 2018.

[34] Stefano Menegat, Alicia Ledo, and Reyes Tirado. Greenhouse gas emissions from global production and use of nitrogen synthetic fertilisers in agriculture. *Scientific Reports*, 12(1): 1–13, 2022.

[35] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, and et al. Human-level control through deep reinforcement learning. *Nature*, 518 (7540):529–533, 2015.

[36] George E Monahan. State of the art—a survey of partially observable markov decision processes: theory, models, and algorithms. *Management science*, 28(1):1–16, 1982.

[37] Hiske Overweg, Herman NC Berghuijs, and Ioannis N Athanasiadis. Cropgym: a reinforcement learning environment for crop management. *arXiv preprint arXiv:2104.04326*, 2021.

[38] Emilio Parisotto, Jimmy Lei Ba, and Ruslan Salakhutdinov. Actor-mimic: Deep multitask and transfer reinforcement learning, 2016.

[39] L. Poggio, L. M. de Sousa, N. H. Batjes, G. B. M. Heuvelink, B. Kempen, E. Ribeiro, and D. Rossiter. Soilgrids 2.0: producing soil information for the globe with quantified spatial uncertainty. *SOIL*, 7(1):217–240, 2021. doi: 10.5194/soil-7-217-2021. URL `https://soil.copernicus.org/articles/7/217/2021/`.

[40] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[41] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021. URL `http://jmlr.org/papers/v22/20-1364.html`.

[42] Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017. URL `https://arxiv.org/abs/1706.05098`.

[43] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[44] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[45] Farooq Shah and Wei Wu. Soil and crop management strategies to ensure higher crop productivity within sustainable environments. *Sustainability*, 11(5), 2019. ISSN 2071-1050. doi: 10.3390/su11051485. URL `https://www.mdpi.com/2071-1050/11/5/1485`.

[46] Shagun Sodhani, Amy Zhang, and Joelle Pineau. Multi-task reinforcement learning with context-based representations. *arXiv preprint arXiv:2102.06177*, 2021. URL `https://arxiv.org/abs/2102.06177`.

[47] Lingfeng Sun, Haichao Zhang, Wei Xu, and Masayoshi Tomizuka. Paco: Parameter-compositional multi-task reinforcement learning, 2022.

[48] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2018.

[49] Ran Tao, Pan Zhao, Jing Wu, Nicolas F. Martin, Matthew T. Harrison, Carla Ferreira, Zahra Kalantari, and Naira Hovakimyan. Optimizing Crop Management with Reinforcement Learning and Imitation Learning, February 2023.

[50] Yee Whye Teh, Victor Bapst, Wojciech Marian Czarnecki, John Quan, James Kirkpatrick, Raia Hadsell, Nicolas Heess, and Razvan Pascanu. Distral: Robust multitask reinforcement learning, 2017.

[51] Matteo Turchetta, Luca Corinzia, Scott Sussex, Amanda Burton, Juan Herrera, Ioannis N. Athanasiadis, Joachim M. Buhmann, and Andreas Krause. Learning long-term crop management strategies with cyclesgym. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. URL `https://openreview.net/forum?id=Zx5qJzNesn0`.

[52] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. URL `https://arxiv.org/abs/1706.03762`.

[53] Nelson Vithayathil Varghese and Qusay H. Mahmoud. A survey of multi-task deep reinforcement learning. *Electronics*, 9(9), 2020. ISSN 2079-9292. doi: 10.3390/electronics9091363. URL `https://www.mdpi.com/2079-9292/9/9/1363`.

[54] Jing Wu, Ran Tao, Pan Zhao, Nicolas F. Martin, and Naira Hovakimyan. Optimizing nitrogen management with deep reinforcement learning and crop simulations. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1711–1719, 2022. doi: 10.1109/CVPRW56347.2022.00178.

[55] Derrick Xin, Behrooz Ghorbani, Ankush Garg, Orhan Firat, and Justin Gilmer. Do current multi-task optimization methods in deep learning even help?, 2022.

[56] Jun Yan and Wang Xiangfeng. Unsupervised and semi-supervised learning: the next frontier in machine learning for plant systems biology. *Wiley Online Library*, 2021. URL `https://onlinelibrary.wiley.com/doi/full/10.1111/tpj.15905`.

[57] Ruihan Yang, Huazhe Xu, Yi Wu, and Xiaolong Wang. Multi-task reinforcement learning with soft modularization. In *NeurIPS*, 2020. URL `https://arxiv.org/abs/2003.13661`.

[58] SUN Yang-yue and SHEN Shuang-he. Research progress in application of crop growth models. *Chinese Journal of Agrometeorology*, 40(07):444, 2019.

[59] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient Surgery for Multi-Task Learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 5824–5836. Curran Associates, Inc., 2020.

Appendix A

Appendix A

A list of input and output files that are used in Cycles simulations. Further information can be found in `psumodeling.github.io/Cycles`

Input files:

- `*.ctrl` - This control file sets the overall parameters for the simulation, including the simulation duration and other settings.
- `*.operation` - This file outlines the agricultural practices and operations to be applied throughout the simulation, such as planting and harvesting dates.
- `*.soil` - This file describes the soil profile, including the properties of each soil layer.
- `*.crop` - This file provides information about the crop species being simulated.
- `*.weather` - This file supplies daily weather data for the simulation, such as min/max temperature, precipitation, wind and humidity.

Output files:

- `environ.txt` - Contains daily environmental data, such as temperature, vapor pressure deficit, and precipitation, which impact crop growth and soil processes.
- `[crop].txt` - Provides information on crop growth, including phenological stage, thermal time, biomass accumulation, and nitrogen content.
- `annualSoilProfileC.txt` - Reports annual carbon dynamics in the soil profile, including residue biomass input, carbon decomposition, and humified carbon.
- `AnnualSOM.txt` - Shows annual carbon concentration and saturation ratio in the soil profile, reflecting soil carbon storage capacity.
- `annualN.txt` - Summarizes annual nitrogen fluxes in the soil profile, including fertilization, fixation, leaching, and denitrification.

- `soilC.txt` - Details carbon dynamics in the soil profile, including microbial biomass carbon, humified carbon, and soil organic carbon.
- `N.txt` - Describes nitrogen dynamics in the soil profile, including soil organic nitrogen, nitrate, ammonium, and nitrogen mineralization.
- `Residue.txt` - Provides information on residue biomass and nitrogen content on the soil surface and in the soil profile.
- `Water.txt` - Reports water balance components, including irrigation, runoff, infiltration, drainage, and soil moisture content.
- `Season.txt` - Summarizes crop harvest data, including total biomass, root biomass, grain yield, forage yield, and nitrogen content.
- `soilLayersCN.txt` - Provides outputs needed to calculate selected C and N pool sizes or elemental concentrations by soil layer.
- `Summary.txt` - Provides a summarized output of total C inputs over the duration of the simulation and average annual rates for N cycling processes.

Appendix B

Appendix B

Monotonicity plots for NHTXx can be found in Figure B.1 and B.2

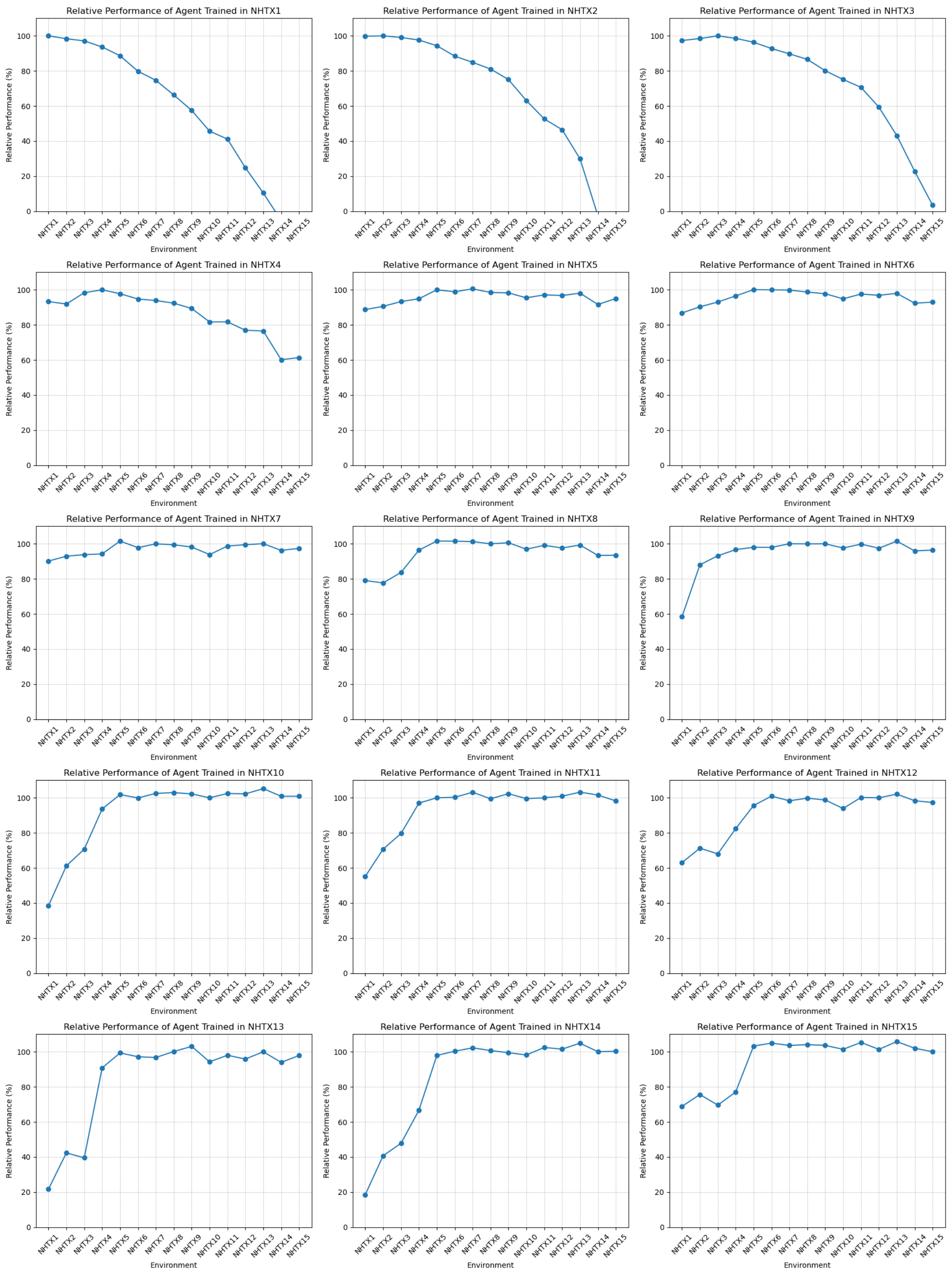

Figure B.1: Performance of each NHTXx-trained agents in each of the NHTXx tasks

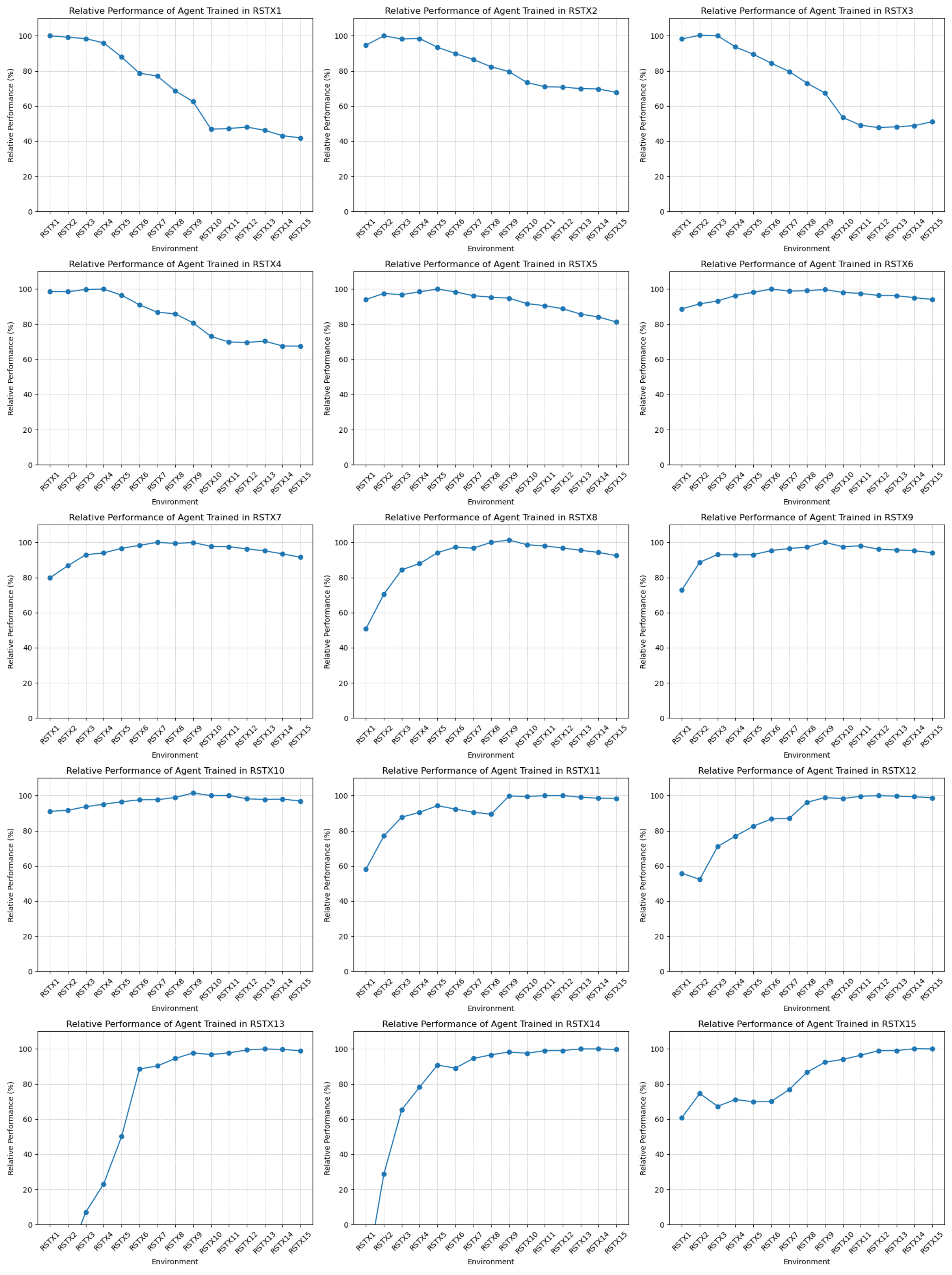

Figure B.2: Performance of each RSTXx-trained agents in each of the RSTXx tasks