OPTIMIZING CROP MANAGEMENT WITH REINFORCEMENT LEARNING, IMITATION LEARNING, AND CROP SIMULATIONS

BY

RAN TAO

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Mechanical Engineering
in the Graduate College of the
University of Illinois Urbana-Champaign, 2022

Urbana, Illinois

Adviser:

Professor Naira Hovakimyan

Abstract

Crop management, including nitrogen (N) fertilization and irrigation management, has a significant impact on crop yield, economic profit, and the environment. Although management guidelines exist, finding the optimal management practices is challenging given a specific planting environment and a crop. This thesis presents an intelligent crop management system that optimizes N fertilization and irrigation simultaneously via reinforcement learning (RL), imitation learning (IL), and crop simulations using the Decision Support System for Agrotechnology Transfer (DSSAT). The thesis formulates the crop management problem as an RL problem, and uses RL algorithms to train management policies that require all state variables from the simulator as observations (denoted as full observation). We then invoke IL to train management policies that only need a limited amount of state variables that can be easily obtained or measured in the real world (denoted as partial observation) by mimicking the actions of the RL-trained policies under full observation. This thesis includes three case studies in total. The first one focuses on optimizing N fertilization under full observation only. The second one optimizes N fertilization and irrigation simultaneously under full observation, and the last one considers N fertilization and irrigation under partial observation. For the case study focusing on N fertilization, experiments are conducted in simulations on the maize crop in Iowa, where farmers usually do not irrigate, and N management policies are trained with two deep RL algorithms, deep Q-network (DQN) and soft actor-critic (SAC). For the case studies on optimizing N fertilization and irrigation simultaneously, we conduct experiments in simulations on maize crop in Florida, where both irrigation and fertilization are critical for the crop growth. Deep Q-network is applied in the RL-based training for finding management policies under full observation, and the RL-trained policies are then used as the expert to train management policies under partial observation with IL. For each case study, we compare the trained policies with baseline methods, which follow a maize production guideline in the corresponding location. The trained policies under both full and partial observations achieve better outcomes than the baseline methods, resulting in a higher profit or yield with less management input or a smaller environmental impact. Moreover, the partial-observation management policies are directly deployable in the real world as they use readily available information, which paves the way for validating the performance of the framework with field tests.

Acknowledgments

I would like to thank my adviser, Professor Naira Hovakimyan, and postdoctoral researcher, Pan Zhao, for letting me join the Advanced Controls Research Laboratory and all their guidance and support since Fall 2021. I would also like to thank all my colleagues. It is my honor to work with these excellent researchers at University of Illinois, and I learned a lot from them on how to become a professional researcher. This work was supported by the C3.ai Digital Transformation Institute and NSF under the RI grant #2133656.

Table of contents

List of Abbreviations

RL	Reinforcement Learning.
IL	Imitation Learning.
DSSAT	Decision Support System for Agrotechnology Transfer.
DQN	Deep Q-network.
SAC	Soft Actor-Critic.
SDM	Sequential Decision Making.
MDP	Markov Decision Process.

Chapter 1

Introduction

The agricultural industry worldwide is facing significant challenges. It needs to produce food for a population expected to reach 9.6 billion by 2050, and simultaneously reduce environmental impacts, including ecosystem degradation and high greenhouse gas emissions [1]. There are plenty of management factors influencing crop yield and environmental impact, among which nitrogen (N) fertilization and irrigation are two of the most significant ones [2]. Based on empirical experience and existing agricultural studies, local best management practices for N fertilization and irrigation exist among farmers. However, it remains to be seen whether the current management practices are optimal and whether these strategies perform well in the presence of changes in climate, yield price, and management cost. Thus, new methods are urgently needed to help farmers build cost-effective and readily deployable systems [3] that provide optimal management policies given a particular condition (including climate, yield price, management cost, etc.) and a target (e.g., maximum economic profit or minimal environmental impact).

Reinforcement learning (RL) has been drawing significant interest in the machine learning and artificial intelligence communities in the last two decades [4]. With RL, the agents learn to perform a task from the outcomes of their own decisions rather than from the decisions of the experts, which makes RL useful in complex problems where finding reliable expert training information is time-consuming, difficult, or even impossible [5]. Sequential decision making (SDM) describes a scenario where consecutive observations of a process are made before a final decision. Considering this, we can view the crop management problem as an SDM problem, where across the growth cycle of the crop, farmers need to obtain information related to the weather, soil, and crop for a period of time in order to make the decision on the management practices, including the amount and time of fertilization and irrigation. Modern reinforcement learning (RL) methods, represented by deep RL, have achieved remarkable or superhuman performance on a variety of tasks involving SDM such as gaming [6], [7], data center cooling [8], marketing and advertising [9], and robotic control [10]–[12]. As a result, we expect that RL has the potential for optimizing agricultural management, improving crop yield while minimizing environmental impacts. Since the RL agent learns from the consequences of their decisions, plentiful interactions between the RL agent and the environment are required for policy training. As real-world farm experiments are laborious, time-consuming, and cost-inefficient, it is impossible to implement field trial-based methods [13] for the training of the management policies, which necessitates the use of agricultural simulation models, including APSIM and DSSAT, for RL-based training [14]. For currently available crop models, the users need to pre-define the management practice before the start of a simulation. However, RL aims to find optimal policies that decide the management practices in (near)

real-time according to the current weather, plant, and soil conditions. Given this concern, efforts have been made to enable real-time communication between an RL agent and the crop environment during a simulation.

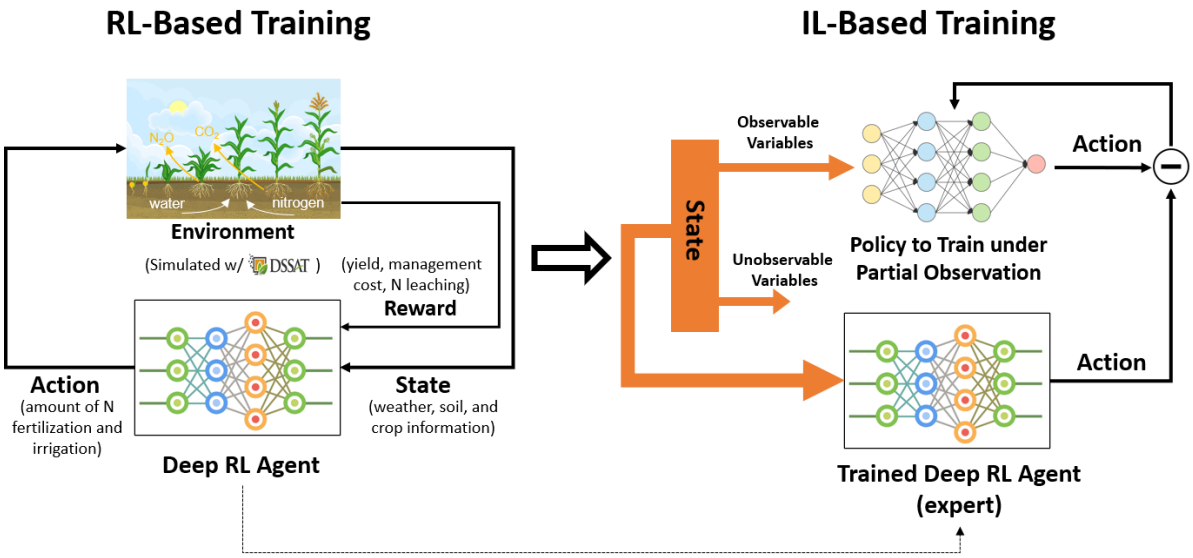

Figure 1.1: Framework of the intelligent crop management system using RL and IL. We first use RL to train management policies under full observation. Then, the RL-trained policies are used as the expert to train policies under partial observation with IL.

This thesis presents an intelligent crop management system, depicted in Figure 1.1, that generates *deployable* and *adaptable* management policies based on RL, imitation learning (IL), and crop simulations via DSSAT. In particular, we leverage Decision Support System for Agrotechnology Transfer (DSSAT), a widely used tool for crop modeling and simulation [15], [16], and the Gym-DSSAT interface [17] that allows users to read the simulated crop and soil conditions and apply management practices on a daily basis. As a demonstration of the use of the presented framework, we conduct three case studies in simulation in total. The first study focuses on the RL-based training under full observation for the maize crop in Iowa, US, where we train N management policies that require all state variables from the simulator as observations with two deep RL algorithms, namely deep Q-network (DQN) and soft actor-critic (SAC). Only N fertilization is included since irrigation is usually not required for maize in Iowa. In the second and third case studies, we train N fertilization and irrigation management policies simultaneously for the maize crop in Florida, US, where both fertilization and irrigation have a significant impact on crop growth. The second case study focuses on RL-based training under full observation and the third one studies IL-based training under partial observation, where the agent only receives state variables that can be easily obtained or measured by farmers in the real world. In particular, DQN is leveraged to train policies under full observation in the second case study, and then the RL-trained policies are used in the third case study as the experts to train policies under partial observation using IL. We further evaluate the performance of the trained policies in comparison with standard practices to validate the benefit of the proposed framework.

Compared to early work on RL-based crop management [18], [19], our framework, which leverages deep RL, can handle much larger state and action spaces. Compared to recent work on deep RL-based agricultural management [20], [21], the crop model adopted in our framework, i.e., DSSAT, is much more widely used globally; additionally, our experimental study is significantly more comprehensive, which involves two different

deep RL algorithms and two geographic locations. Additionally, in the second case study, we investigate the RL-based policy training with different reward functions that represent different tradeoffs among crop yield, N fertilizer use, water use, and environmental impact during the crop growth cycle. We also analyze the adaptation of the trained policies when a different target, represented by the reward function, is provided. Most importantly, we leverage IL as a new tool to find the optimal management policies that require only state variables that can be easily obtained or measured in the real world. As a result, the path to deployment of our intelligent crop management system is paved, and field tests can be conducted to prove the effectiveness of our trained policies.

The thesis is organized as follows. In Chapter 2, we present some work related to the topic of applying RL and IL for finding optimal crop management policies. Chapter 3 discusses the details of the method we used for the application of RL and IL. Chapters 4-6 present the results and analysis of three case studies from the proposed framework. Chapter 7 discusses the path to deployment of our framework and some potential problems regarding the sim-to-real issue. Finally, Chapter 8 concludes the thesis and presents some possible future work.

Chapter 2

Related Work

2.1 Reinforcement Learning in Crop Management

Reinforcement learning, as a sub-field of machine learning, aims to solve sequential decision making (SDM) problems by letting an agent directly interact with the environment and learn from trial and error [22]. By viewing crop management as an SDM problem, to be more specific, as a Markov decision process (MDP) problem, researchers have applied RL to find the optimal crop management policies from simulators. As a pioneering work, [18] proposed to use a simple RL method (namely, R-learning) and crop simulations to optimize management of wheat crops in France. [19] studied the use of SARSA(λ), an on-policy RL method, and crop simulations to optimize the irrigation for the maize crop in Texas, US. However, the state and action spaces in [18], [19] were quite small due to the curse of dimensionality from which early RL methods suffered. For instance, the state space in [19] has only one state, i.e., total soil water (TSW) level. In contrast, modern RL methods, represented by deep RL, are able to handle extremely large state and action spaces due to the use of deep neural networks (DNNs) (to approximate the value functions or policies), and have achieved remarkable or superhuman performance on a variety of high-dimensional problems. Deep RL based on the proximal policy optimization (PPO) algorithm was used in [20] to optimize the fertilizer management for the wheat crop. Additionally, [21] studied the use of PPO to optimize the irrigation management for russet potatoes. However, the study is quite coarse and the results are not promising. For instance, in terms of results, [21] included only a simple learning curve showing the normalized reward, while the variables farmers mostly care such as yield, management cost, and nitrate leaching were not included. Additionally, the trained policy performed much worse than a simple policy which applies a constant amount of water.

2.2 Crop Models for Reinforcement Learning

Crop models can simulate crop growth in response to soil, water, nutrient, and weather dynamics. They are playing increasingly important roles in the development of sustainable agricultural management, because field and farm experiments require large amounts of resources and may still not provide sufficient information in space and time to identify appropriate and effective management practices [23]. The development of crop models dates back to 1950s. In the past seven decades, many crop models of varying complexities have been developed by different groups, which include Agricultural Production Systems Simulator (APSIM), CERES (now contained in the DSSAT Suite of crop suite), CROPSYST, EPIC, WOFOST, and COUP. See the survey

thesis [23] and a comparison of different crop models for yield response prediction [24]. Among the existing crop models, the ones that are extensively used globally are APSIM and DSSAT, which are still constantly evolving and currently open-source to facilitate community-based development.

Most of the existing crop models need the management practices to be pre-specified before the start of a simulation, while RL-based training of management policies requires the management practices to be determined according to the soil, plan and weather conditions on a daily or weekly basis during the simulation. In light of this, the authors of [20] developed the CropGym environment for training of N management policies, which provides an interface to Open AI Gym [25], a widely used toolkit for RL research, and enables an RL agent to interact with the crop environment weekly. However, CropGym is based on the LINTUL-3 model [26] for the wheat crop, which has limited use. In a similar spirit, [21] presented another crop environment with the Open AI Gym interface for the russet potato based on the SIMPLE crop model [27], which, again, has limited use, potentially because the model is over-simplified. Recently, a Gym-DSSAT environment for the maize crop, which is based on the widely used DSSAT suite of crop models and provides a Gym interface, was developed [17] and enables an RL agent to interact with the environment on a daily basis. However, there have been no results on the use of Gym-DSSAT for training crop management policies up to now.

2.3 Imitation Learning for Policy Training

IL seeks to learn a policy by imitating the behavior of an expert and it has been widely applied in the field of robotics, including autonomous aerial and ground vehicles. The authors of [28] applied IL to learn autonomous driving policies using an expert driver as the learning target. The authors of [29] alternatively used the model predictive controller (MPC) as the expert to train a control policy under partial observation. Both of these works indicate the promising ability of IL in learning good policies given an expert. [29] further demonstrates the ability of IL in learning good policies under partial observation, which is usually difficult to achieve with standard RL due to the less available information compared to the full observation case. However, to the best of our knowledge, there has been no reported work on applying IL for crop management. Thus, from [29], we adopt the idea of using IL to solve the problem of partial observation given an expert policy under full observation, which is summarized as the IL-based learning in the proposed framework in Figure 1.1.

Chapter 3

Methods

We now present technical details for the framework based on deep RL, IL, and crop simulations depicted in Figure 1.1.

3.1 Markov Decision Process Problem Formulation

The crop management is formulated as a finite Markov decision processes (MDP) here. In this formulation, a decision-making agent continuously interacts with the environment. On each day t, the agent receives the state of the environment, s_t, and chooses the action a_t from the action space $\mathcal{A}$ based on some policy $\pi(s_t, \theta_t)$, where θ_t is the parameter of the policy at current day. The state s_t contains information related to the weather, plant, and soil at the given day from the simulator.

For the first case study of maize in Iowa, action a_t only consists of the amount of N fertilizer to be applied for that day, N_t. The reward function $r_t(s_t, a_t)$ at day t is set as:

$$r_t(s_t, a_t) = \begin{cases} w_1 Y - w_2 N_t - w_3 N_{l,t} - w_4 P_t & \text{if harvest at } t, \\ -w_2 a_t - w_3 N_{l,t} - w_4 P_t & \text{otherwise,} \end{cases} \tag{3.1}$$

where w_1, w_2, w_3, w_4 are four weight factors to be determined, $N_{l,t}$ is the nitrate leaching at day t, Y is the crop yield at the harvest date, and P_t is the additional penalty on large *total amount* of nitrogen applied. In particular, the penalty P_t is designed as:

$$P_t = \begin{cases} \sum_{k=1}^{t} a_k - threshold & \text{if } a_t \neq 0, \\ 0 & \text{if } a_t = 0, \end{cases} \tag{3.2}$$

where the *threshold* represents the allowable total amount of nitrogen inputs to be decided. The inclusion of P_t in the reward function of the first case study is optional since we already have w_2 in the reward function to penalize the use of fertilizer. We include P_t to help the training converge faster. Also, the total amount of N fertilizer decided from the trained policy will be smaller than the *threshold* in order to achieve maximum cumulative reward, which indicates that P_t will not influence the evaluation of the trained policies. It may be worth mentioning that nitrate leaching occurs when nitrate is washed out of the root zone by heavy rainfall or irrigation. Leaching is undesirable because it leads to the waste of the fertilizers, and more importantly, causes environmental problems such as eutrophication of watercourses and soil degradation. Thus, we include

a penalty on nitrate leaching in the reward function.

For the second and third case studies of maize in Florida, a_t now consists of both N_t, the amount of N fertilizer input, and W_t, the amount of irrigation water to be applied for day t. Similarly, given s_t and a_t, the reward function $r_t(s_t, a_t)$ for these two case studies is defined as:

$$r_t(s_t, a_t) = \begin{cases} w_1 Y - w_2 N_t - w_3 W_t - w_4 N_{l,t} & \text{if harvest at } t, \\ -w_2 N_t - w_3 W_t - w_4 N_{l,t} & \text{otherwise,} \end{cases} \tag{3.3}$$

where $w_1, w_2, w_3, w_4, Y, N_{l,t}$ are four weight factors to be determined, yield at harvest, and the amount of nitrate leaching at given day respectively.

Given the state s_t and a design of the reward function represented by w_1, w_2, w_3, w_4, the goal of the agent is to find the optimal policy $\pi(s_t, \theta_t)$ which renders a_t and maximizes the future discounted return, which is defined as:

$$R_t = \sum_{\tau=t}^{T} \gamma^{\tau - t} r_\tau, \tag{3.4}$$

which represents the sum of the reward at current day from applying a_t, i.e., r_t, and discounted future rewards with a discount factor γ following this policy.

3.2 Training Management Policies under Full Observation using Deep RL

For solving the formulated MDP problem, we leverage the recently proposed deep RL algorithms, which have achieved remarkable performance on a variety of tasks [6]–[12]. We choose deep Q-network (DQN) [6] and soft actor-critic (SAC) [30] for the experimental study, but other deep RL algorithms can also be applied. Due to the nature of different RL algorithms, the action space used in DQN is discrete, while the action space used in SAC is continuous.

3.2.1 Policy Training with DQN

With DQN, a deep neural network is used to represent the action-value function, also known as the Q function [6], and thus we call the network as a Q-network. The Q function of a policy π is defined as:

$$Q^\pi(s, a) = \mathbb{E}[R_t | s_t = s, a_t = a, \pi], \tag{3.5}$$

which measures the expected future discounted return obtained from state s by taking action of a and following policy π afterwards. We keep updating the parameters of the Q-network to find the optimal Q function, $Q^\star(s, a)$, which represents the optimal return that can be gained from state s by taking action a and following the optimal policy afterwards. Given an optimal Q function, $Q^\star$, and a state s_t, a greedy policy defined as:

$$a_t^\star = \max_{a \in \mathcal{A}} Q^\star(s_t, a) \tag{3.6}$$

is used by the agent to determine the optimal action. Since the Q function determines the policy of the agent, training of the Q-network is same as the training of the policy. The Q-network parameter at iteration i, denoted by θ_i, is updated by minimizing the loss function:

$$L_i(\theta_i) \triangleq \mathop{\mathbb{E}}_{(s,a,r,s')} \left[r + \gamma \max_{a' \in \mathcal{A}} Q(s', a'; \theta_i^-) - Q(s, a; \theta_i) \right], \tag{3.7}$$

where s, a, r, s' and γ denote current state, current action, current reward of s and a, next state, and discount factor respectively, and θ_i^- denotes the parameters of a previously defined target network. The tuples (s, a, r, s') are randomly chosen from the replay buffer, which is a memory base of previously generated tuples (s, a, r, s') during training.

3.2.2 Policy Training with SAC

SAC is a policy-gradient deep RL algorithm that represents the state of the art among model-free RL algorithms in terms of sample efficiency and stability with respect to the hyperparameters [30]. Besides the expected future discounted rewards, SAC introduces the expected entropy to favor stochastic policies, which leads to a cost function L defined as:

$$L \triangleq -\sum_{t=0}^{T} \mathbb{E}_{(s_t, a_t) \sim p_\pi} \left[r_t(s_t, a_t) + \alpha \mathcal{H}(\pi(\cdot|s_t)) \right], \tag{3.8}$$

where p denotes the state-action marginals of the trajectory distribution, $\mathcal{H}$ determines the entropy for the evaluation of randomness given the state s_t, and $r_t(s_t, a_t)$ is the immediate reward at time t given state s_t and action a_t. The temperature parameter α decides the trade-off between the entropy term and rewards.

3.3 Crop Simulations and Daily Interactions with DSSAT and Gym-DSSAT

DSSAT has been used for various crop simulations worldwide in the last 30 years [15]. However, limited interactions can be reached during the running period of simulation, leading to a possible delay of adjustment for management decisions. Recently, Gym-DSSAT [17] has been developed to bridge the communication gap between the simulation environment and daily management decisions. This communication pipeline enables RL researchers to manipulate DSSAT like Open AI Gym in machine learning and robotics [6], [7]. In Gym-DSSAT, the environment is defined at a field scale with a time step corresponding to one day. An episode typically covers about 160 days from planting to harvest, and its state is automatically set as "done" at crop maturity. Weather is randomly generated via WGEN's[31] built-in stochastic weather generator and can be fixed depending on simulation purposes.

With Gym-DSSAT, millions of daily interactions between an RL agent and the simulated crop environment can be achieved in a few minutes, and used for training the management policies.

3.4 Training Management Policies under Partial Observation using Imitation Learning

Imitation learning aims to train the agent to perform a task by mimicking the behavior of an expert [32]. As opposed to learning from scratch by trial-and-error in RL [33], with IL the agent learns a mapping between the observations and desired actions determined by the expert, which simplifies the learning process of complex problems. For the crop management problem, not all state variables from the simulator can be achieved or measured by farmers. Thus, for deployment in the real world, the agent should only utilize state variables that are accessible to farmers. Given any state s, denote s^o as the observable state which contains variables from s that are observable or measurable by farmers. Under partial observation, on each day t, the agent receives s_t^o. The goal of the agent is to learn an optimal policy $\pi(s_t^o, \theta)$ that generates an action a_t^o that is same as a_t, where a_t is the action determined from the expert given an observation of s_t. Behavior cloning, the simplest form of IL, can be applied to train the policy under partial observation as follows. We first collect demonstrations, state-action pairs (s, a), from the expert policy and store them into a dataset $\mathcal{D}$. Then, the policy network of the agent is updated by minimizing the loss function:

$$L(\theta) = \sum_{(s,a)\in\mathcal{D}} \|\pi(s^o, \theta) - a\|. \tag{3.9}$$

The loss function represents the difference between the output of the policy network with s^o as the input, and the action a determined by the expert policy given s.

Chapter 4

Optimizing N Fertilization under Full Observation

In this chapter, we present the experiment setup and results for the case study of optimizing N fertilization under full observation for maize crop in Iowa, where farmers usually do not irrigate for maize. We leverage both DQN and SAC to train the RL agent, and test the performance of all the trained policies in simulation and compare their results with baseline policies proposed in [34].

It is worth mentioning that all experiments presented in Chapters 4-6 aim to validate the feasibility and benefit of the proposed framework, not for the immediate deployment. For real world application of our framework, current soil and weather data corresponding to the testing farm needs to be collected to configure the crop model within DSSAT, and more details can be found in Chapter 7.

4.1 Dataset for Simulation

The experiments in this chapter simulate the growth of the maize crop in Ames, Iowa, in 1999. The simulation starts on April 25th, the planting happens on May 27th, and the crop is harvested no later than Oct 24th. The soil has a depth of 151 cm, and the plant density is 7.6 plant/m^2. For this case study, the irrigation is set to 0. This is consistent with the current practice in Iowa, where the maize crop is not irrigated. The detailed description of state variables from the simulator can be found in Table A.1 in Appendix A. The details of the baseline polices are described below. The baseline policy applies 160, 240, or 280 kg/ha of N fertilizer when the crop reaches stage v5, vegetative stage 5.

4.2 Implementation Details

For all the experiments in this case study, weight parameters w_1, w_2, and w_3 in the reward function (3.1) were set to be 0.1, and w_4 was set to be 1. We chose to use the state variable "topwt" (top weight (kg/ha)) from Table A.1 to represent the yield Y in the reward function (3.1). For both DQN and SAC, we implemented the training using Pytorch, and used the Adam [35] optimizer with an initial learning rate of 0.00005 and a batch size of 64 to train the neural network. The deep neural network was designed to have 2 hidden layers with 64 hidden units in each layer. We trained the policies for 1200 episodes with the exploration rate ϵ decreasing

from 1 to 0, following a decay factor of 0.992. For DQN, the discrete action space was defined to be:

$$\mathcal{A} = \{40k \frac{\text{kg}}{\text{ha}} \text{ N fertilizer} | k = 0, 1, 2, 3, 4\}. \tag{4.1}$$

This design of the action space includes standard amounts of N fertilizer that farmers can potentially apply in a single day and also provides enough options for finding good policies. The discount factor was set to be 0.99. For SAC, the agent action a_{sac} varies from 0 to 200 and is discretized into the same action space as the one used for DQN through the mapping:

$$\arg\min_{a \in \mathcal{A}} \quad \|a_{sac} - a\|, \tag{4.2}$$

for both training and testing. The discretization is included for being consistent with the action space used in the training using DQN and also with farmers' fertilization patterns, i.e., fertilize only a few times in the whole growth cycle. The discount factor and smoothing constant for updating the target network were set to be 0.98 and 0.001, respectively.

For comparison with the trained policies, we also implemented the standard management practice in [34], which suggests to add nitrogen at vegetative growth stage (vstage) 5, the stage when crop reaches five expanded leaves.

4.3 Training Results and Evaluations

The training curve using DQN, averaged over five trials, is shown in Fig. 4.1. During the first 200 episodes of exploration, due to the large exploration rate, the DQN agent over-fertilizes, causing significant penalties due to the existence of P_t in (3.1). After 800 episodes of training, the learning converges, constantly giving a cumulative reward of over 2000. Concretely, Table 4.1 compares the performance of the DQN-trained policy and three baseline strategies, corresponding to 160, 240, 280 kg/ha of nitrogen applied at stage v5, as suggested in [34]. The trained DQN agent decides to apply a total of 240 kg/ha nitrogen fertilizer input during the growing season, and achieves 21711.8 kg/ha top weight of maize at maturity, 0.11 kg/ha of nitrate leaching, and a cumulative reward of 2147.1. Among three baseline policies, the one with 280 kg/ha achieves the largest cumulative reward of 2142.9 and largest top weight of 21709.5 kg/ha. Compared with the best baseline, DQN achieves slight improvement on top weight (yield) at maturity while using 14% less nitrogen input, being more cost-efficient. Compared to the baseline using same amount of nitrogen input, DQN achieves a 1% increment on the top weight (yield) at harvest. In general, the trained DQN agent achieves better results than the baseline methods.

Table 4.1: Performance comparison between the trained policy from DQN and baseline policies for Iowa. Baseline (X) indicates that X kg/ha of nitrogen is applied at stage v5. The trained policy from DQN achieves a higher cumulative reward than the baseline methods.

Methods	Nitrogen input (kg/ha)	Nitrate leaching (kg/ha)	Top weight at maturity (kg/ha)	Cumulative reward
Baseline (160)	160	0.11	21133.3	2097.3
Baseline (240)	240	0.11	21502.9	2126.3
Baseline (280)	280	0.11	21709.5	2142.9
DQN	240	0.12	**21711.8**	**2147.1**

The performance of the trained policy from SAC is shown in Table 4.2. Although using less nitrogen

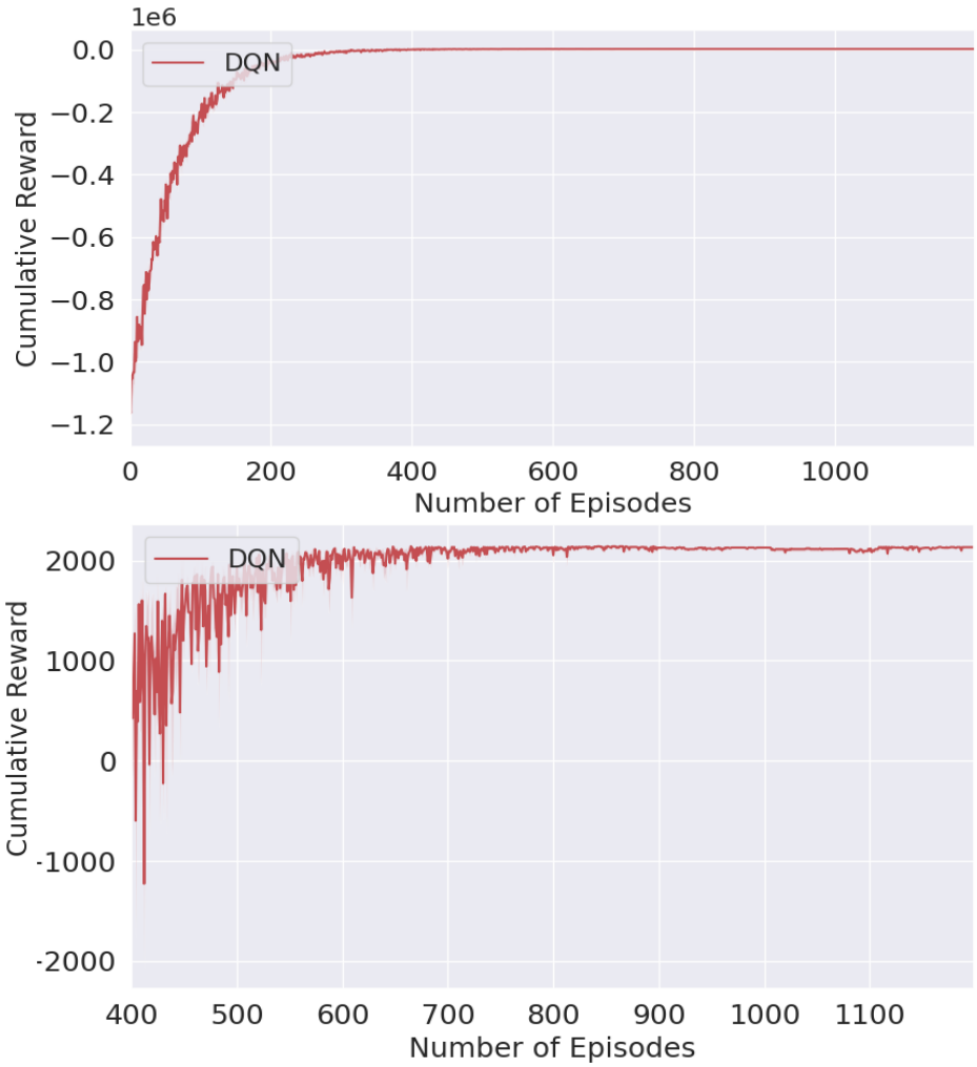

Figure 4.1: Cumulative reward versus episodes for training with DQN for Iowa. Results are averaged over five trials, with the light-red shaded area denoting the variance. Top: full view. Bottom: zoomed-in view for 400–1200 episodes

that causes a smaller top weight at maturity, the trained policy from SAC still achieves a cumulative reward similar to that achieved by the trained policy from DQN. Thus, both RL algorithms succeeded in finding a better management policy than the baseline. However, the learning process converged much faster with SAC. Specifically, the cumulative reward of the training curve with SAC reached 2100 within 700 episodes, while additional 300 episodes were needed to achieve similar results with DQN.

Table 4.2: Performance comparison between trained policies from SAC and DQN for Iowa. The trained policies from different RL algorithms achieve similar cumulative rewards and outperform the baseline method.

Methods	Nitrogen input (kg/ha)	Top weight at maturity (kg/ha)	Cumulative reward	Episodes of convergence
Baseline (280)	280	21709.5	2142.9	N/A
DQN	240	21711.8	2147.1	1000
SAC	200	21503.3	2144.2	700

Chapter 5

Optimizing N Fertilization and Irrigation under Full Observation

In this chapter, we present the experiment setup and results for the case study of optimizing N fertilization and irrigation simultaneously under full observation for maize crop in Florida, where irrigation is crucial for the growth of maize crop. We leveraged DQN to train the RL agent with 5 different reward functions to demonstrate the adaptability of our framework to different tradeoffs among crop yield, N fertilizer use, irrigation water use, and environmental impact. The performance of all the trained policies was compared with a baseline policy following a corn production guideline for farmers in Florida from [36].

5.1 Dataset for Simulation

The experiments in this chapter simulate the growth of the maize crop in Gainesville, Florida, in 1982. The simulation starts on Jan 30th, while the crop is planted on Feb 26th and harvested when reaching maturity. The soil in this case has a depth of 180 cm, and the plant density is 7.2 plant/m^2. The details of the baseline policy are described below. For N fertilization, 40 kg/ha of fertilizer is applied at the planting date, 40 kg/ha is applied when the maize reaches about 12 inches tall, 150 kg/ha is applied 4 weeks after, and finally 130kg/ha is applied 6 weeks after. For irrigation, one inch of water is fed every 10 days until the maize reaches 15 inches high, an inch is applied every 7 days before tassel emergence, and finally an inch of water is applied every 3 days until maize maturity.

5.2 Implementation Details

The neural network of the Q function defined in (3.5) was designed to have 3 hidden layers with 256 units in each layer. The discrete action space was set as:

$$\mathcal{A} = \{40k\frac{\text{kg}}{\text{ha}} \text{ N fertilizer } \& \ 6k\frac{\text{L}}{m^2} \text{ Irrigation water} | k = 0, 1, 2, 3, 4\}, \tag{5.1}$$

with a size of 25. This design of the action space includes standard amounts of N fertilizer and irrigation water that farmers can potentially apply in a single day and also provides enough options for finding good policies. The discount factor was set to be 0.99. For updating the neural network, we utilized Pytorch and

Adam [35] optimizer with an initial learning rate of 1e-5 and a batch size of 640. We trained the policies for 4000 episodes with the exploration rate ϵ decreasing from 1 to 0, following a decay factor of 0.994.

Five different functions for r_t in (3.3) were used to train the RL agent and we obtained five different trained policies. The parameters used in each reward function (RF) are listed in Table 5.1. RF 1 represents the economic profit ($/ha) that farmers gain based on the approximate price of maize and cost of N fertilizer and irrigation water from [37] and [36]. RFs 2-4 indicate the economic profit under different situations. To be more specific, RF 2 represents the extreme case when irrigation water is free; RF 3 denotes the extreme case when N fertilizer is free, and RF 4 simulates the situation when the price of N fertilizer is doubled. Compared with RFs 1-4 that consider economic profit only, RF 5 includes the additional term of nitrate leaching, an environmental factor. RF 5 is designed with similar weights on Yield, N fertilizer usage and irrigation usage, and a much larger weight on nitrate leaching to promote minimal nitrate leaching while obtaining a good economic profit. In agriculture, both the grain weight and top weight can be used to represent yield of the crop. Here, we choose to use the state variable "grnwt" (grain weight dry matter (kg/ha)) from Table A.1 to represent the yield N_t in the reward function, since the approximate price of maize is based on the grain weight of maize instead of top weight.

Table 5.1: Parameters used in each reward function (RF) defined by (3.3)

	w_1	w_2	w_3	w_4
RF 1	0.158	0.79	1.1	0
RF 2	0.158	0.79	0	0
RF 3	0.158	0	1.1	0
RF 4	0.158	1.58	1.1	0
RF 5	0.2	1	1	5

Table 5.2: Evaluation results of trained policies under full observation and the baseline policy. N_l represents the N leaching amount. Trained Policy x indicates the training result of the RL agent using reward function (RF) x. The trained policies from our RL-based framework have different strategies targeting their own reward function design.

	N Input (kg/ha)	Water Input (L/m^2)	N_l (kg/ha)	Yield (kg/ha)
Baseline Policy	360	393.7	212.6	10771.5
Trained Policy 1	200	120	35.5	10852.4
Trained Policy 2	200	732	59.4	11243.8
Trained Policy 3	19920	108	6205.0	10865.2
Trained Policy 4	160	102	34.9	10357.6
Trained Policy 5	200	138	39.2	10926.1

5.3 Training Results and Evaluation

Five trained policies were achieved using five different reward function designs. For illustration purpose, the training curves using RF 1, which only considers economic profit, and RF 5, which includes both economic and environmental considerations, are shown in Figure 5.1. According to Figure 5.1, we see that the initial cumulative rewards for both of the training curves are extremely small due to the initial large exploration rate, which leads to a large amount of fertilization and irrigation. The initial cumulative reward of the training curve using RF 5 is even smaller due to the additional penalty of nitrate leaching. Then, the training

curve using RF 1 converges after around 2500 episodes, reaching a cumulative reward of about 1600, and the training curve using RF 5 also converges after 2500 episodes, but with some fluctuations. The reward function design in this case study (3.3) does not include the additional penalty P_t, which was included in (3.1), and both training curves converge. Thus, the inclusion of P_t is optional in the design of the reward function in order to find the optimal management policies.

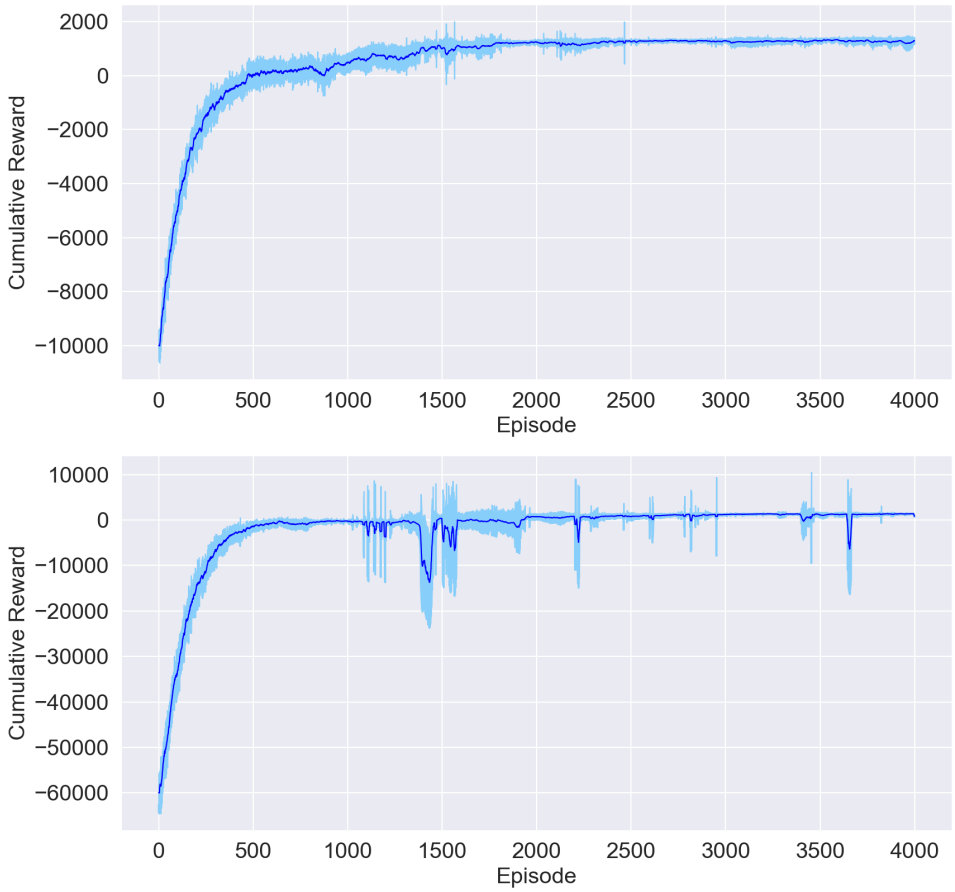

Figure 5.1: Cumulative reward versus episodes for policy training under RF 1 (Top) and RF 5 (Bottom). Results are averaged over five trials, with light-blue shaded areas denoting the variance.

The evaluation results of all the trained policies under full observation are shown in Table 5.2 and Table 5.3. It is worth mentioning that due to the random initialization of the Q-network and the fact that the Q-network gets updated every episode, the policies evaluated above may not represent the most ideal one from training. However, the trained policies we picked are representative enough to demonstrate the ability of RL to optimize crop management and the influence of the reward functions on the training results. According to Table 5.2, the reward function affects the strategy of the trained policy significantly. For example, trained with RF 2 which indicates the zero cost of irrigation water, the Trained Policy 2 applies the biggest amount of irrigation water while keeping the N input low, leading to the highest yield and the largest cumulative reward according to RF 2. Similarly, with a zero cost of nitrogen fertilizer in RF 3, the Trained Policy 3 applies a large amount of N fertilizer with a small amount of irrigation. Regrading RF 4 when the cost of N fertilizer doubles, the Trained Policy 4 results in a slightly smaller amount of N input with a similar amount of water input. According to Table 5.3, we see that given a RF to compute the cumulative rewards

Table 5.3: Performance of the baseline policy and trained policies in terms of cumulative reward computed using different reward functions (RFs). For each RF, the largest cumulative reward value is shown in bold.

	RF 1	RF 2	RF 3	RF 4	RF 5
Baseline Policy	984	1417	1269	700	338
Trained Policy 1	**1425**	1557	1538	1267	**1673**
Trained Policy 2	813	**1619**	971	655	1020
Trained Policy 3	-14139	-14020	**1598**	-29876	-48880
Trained Policy 4	1398	1510	1524	**1272**	1635
Trained Policy 5	1417	1568	1575	1259	1651

of different trained policies, the largest reward is almost always achieved by the policy trained with this particular RF (e.g., Trained policy 1 achieves the highest cumulative reward with RF 1). For RF 5, Trained policy 5 achieves a cumulative reward slightly smaller than the largest one from Trained Policy 1 but still much larger than the one from the baseline policy. The enormous negative values of the cumulative rewards of Trained Policy 3 are caused by the extremely large amounts of N input, which are not punished during the training of Trained Policy 3 using RF 3. In general, the results above demonstrate the ability of RL to optimize crop management under different criteria.

Application history of N fertilizer and irrigation water from all the trained policies are also analyzed. For illustration purpose, the application history of the baseline policy and Trained Policy 1, and the history of the baseline policy and the Trained Policy 5 are visualized in Figure 5.2 and Figure 5.3 respectively. For all of the trained policies, most of the N fertilizer and irrigation water are applied during April to June, which is the crucial growth period for maize [38]. This fact further certifies the reasonableness of our trained policies and thus our proposed framework.

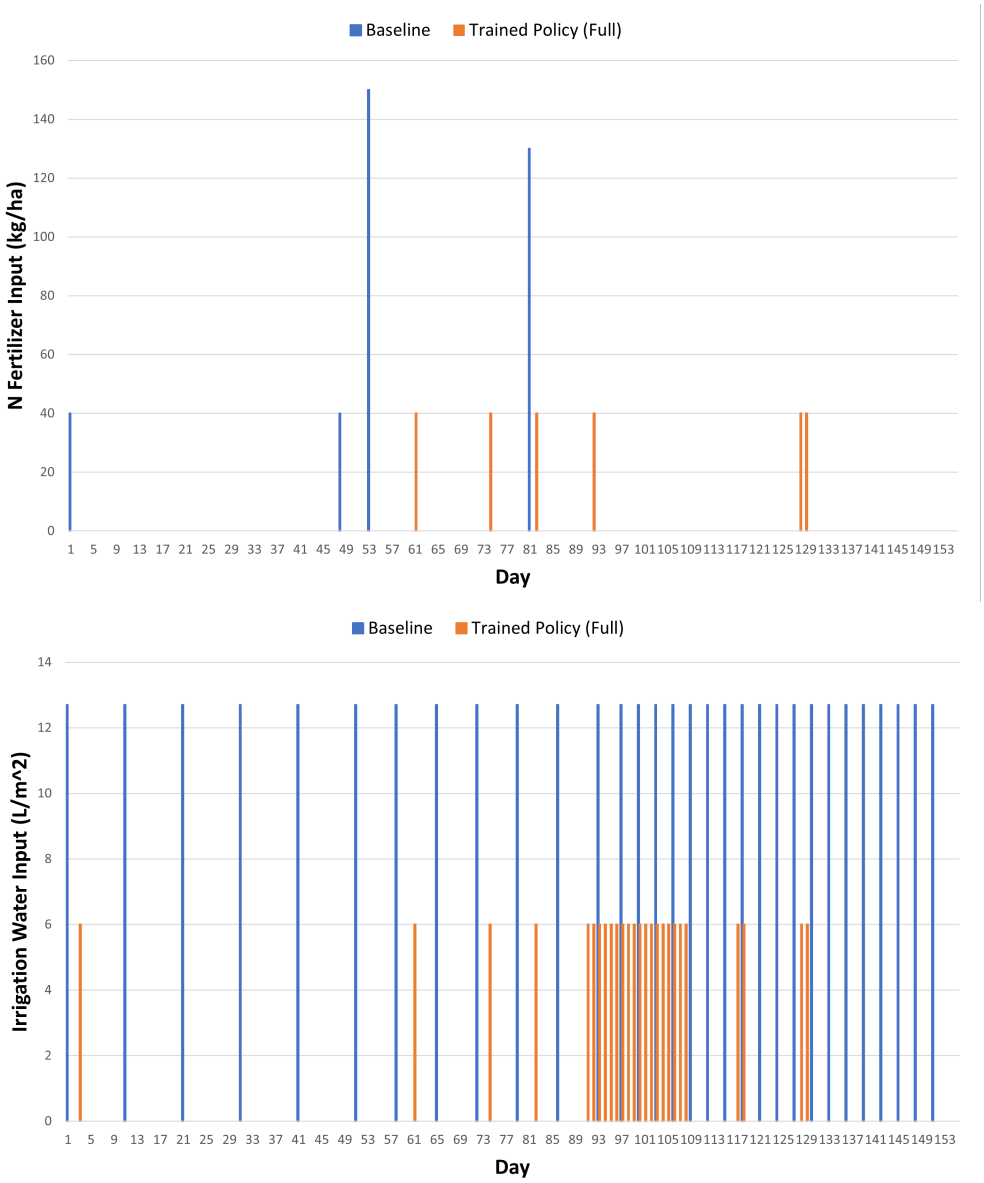

Figure 5.2: Comparison of the N fertilization (Top) and irrigation (Bottom) determined from the baseline policy and Trained Policy 1.

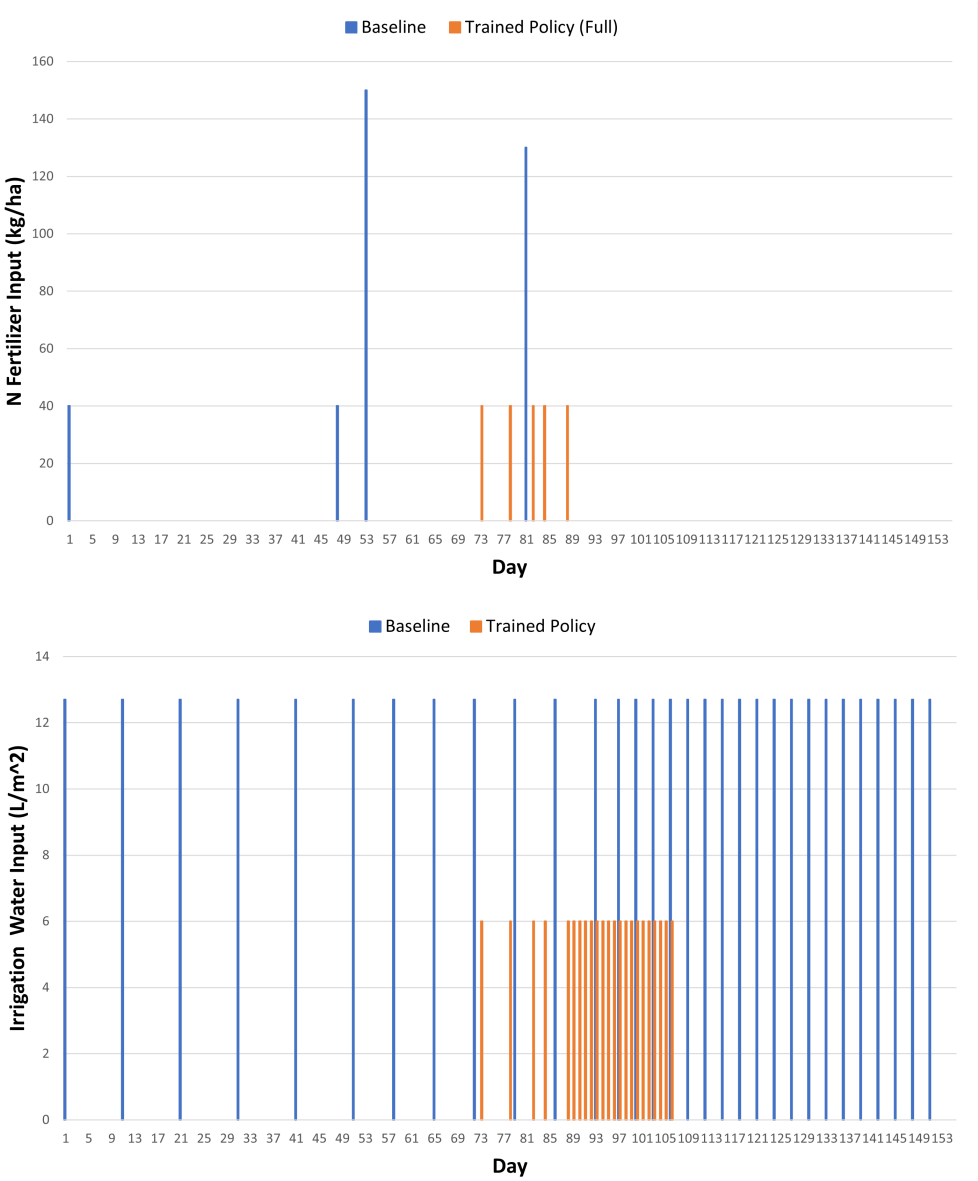

Figure 5.3: Comparison of the N fertilization (Top) and irrigation (Bottom) determined from the baseline policy and Trained Policy 5.

Chapter 6

Optimizing N Fertilization and Irrigation under Partial Observation

During the policy training under full observation, we included all the state variables from DSSAT as the input to the RL agent, and thus the neural network. However, most of the state variables available in DSSAT, including nitrate leaching and daily nitrogen denitrification, cannot be obtained or even measured by farmers with existing instruments in reality. As a result, there is no way to implement those trained policies on a real farm, and a policy that only utilizes state variables that can be easily obtained or measured (partial observation) is necessary for field tests.

In this chapter, we present the experiment setup and results for the case study of optimizing N fertilization and irrigation simultaneously under partial observation for maize crop in Florida. We first tried to train the management policies under partial observation using standard RL, but achieved undesirable results. Then, we utilized IL to train the policies under partial observation using the previously RL-trained policies under full observation as the expert to imitate from. For the state variables used in the partial observation study, all of them can be easily accessed or measured by farmers in the real world, like the weather information, soil water content, which can be measured by a soil moisture meter, and maize growing stage, which can be easily identified by an experienced farmer. The detailed state variables included in the partial observation study can be found in Table A.1.

6.1 Policy Training with RL

Under partial observation, we first experimented on the policy training with RL, and the setup was identical to the case of full observation, except that the state space used here was smaller since only accessible state variables were included. Although we tried policy training using RL with different dimensions of the neural network, learning rates, exploration decay rates, and batch sizes, all trained policies converged to a single one which applies zero N fertilizer and zero irrigation water every day. A typical training curve for policy training under partial observation with RL is shown in Figure 6.1. Based on this figure, the cumulative reward of the training curve from RL under full observation keeps increasing during the first 2000 episodes, while the cumulative reward of the training curve from RL under partial observation easily converges to a small value and stays. Thus, it is much challenging to find a good policy under partial observation with standard RL, which works well for policy training under full observation. The unsatisfactory training results from RL

under partial observation motivate us to leverage IL, a straightforward approach for policy training under partial observation given an expert, which can be the previously RL-trained policies under full observation. There are other commonly used techniques to deal with partial observation in RL like including the history of observations and actions or use predictive rewards [39]. However, they may require much more effort and work, and thus we consider IL as the best solution.

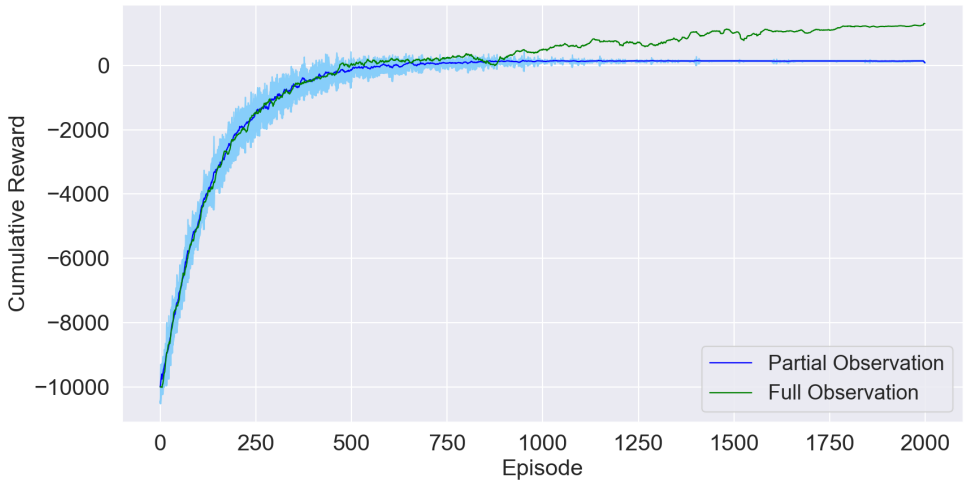

Figure 6.1: Cumulative reward versus episodes for policy training under partial observation with RF 1. Results are averaged over five trials, with the light-blue shaded area denoting the variance. The cumulative reward achieved in the partial observation case is much smaller than that from the full observation case.

6.2 Policy Training with IL

With IL, the policies were trained by imitating the actions determined by the experts, which are the previously RL-trained policies under full observation. We ran the simulations with the expert policies to create the demonstrations and stored the generated pairs of observation and action into a memory base. Mean squared error loss function was applied and stochastic gradient descent algorithms were used to minimize the loss function defined in (3.9).

We conducted two experiments with IL, one using Trained Policy 1 as the expert, and the other with Trained Policy 5. A deep neural network with 3 hidden layers and 256 hidden units was used to represent the policy, which is same as the setup in full observation study but with different dimensions of the input layer and output layer. The input layer has a smaller dimension due to a fewer number of input features, and the output of the network is designed to contain only two elements, representing the amount of N fertilizer and irrigation water respectively. A Sigmoid function was used at the last layer of the network to restrict the range of the output such that the first element is between 0-160 and the second element between 0-24. Thus, the range of the output in the partial observation study using IL is same as the range of the output defined in the full observation study using RL. The batch size of the stochastic gradient descent for policy training under RF 1 was set to 64, and 10 for the policy training under RF 5. The training curves for both experiments are shown in Figure 6.2. The loss of the training curve under RF 1 converges to almost 0 after 20000 steps, and the loss of the training curve under RF 5 converges to almost 0 after 80000 steps due to a smaller batch size. Note that the policy to be trained here has a continuous action space, while RL-trained

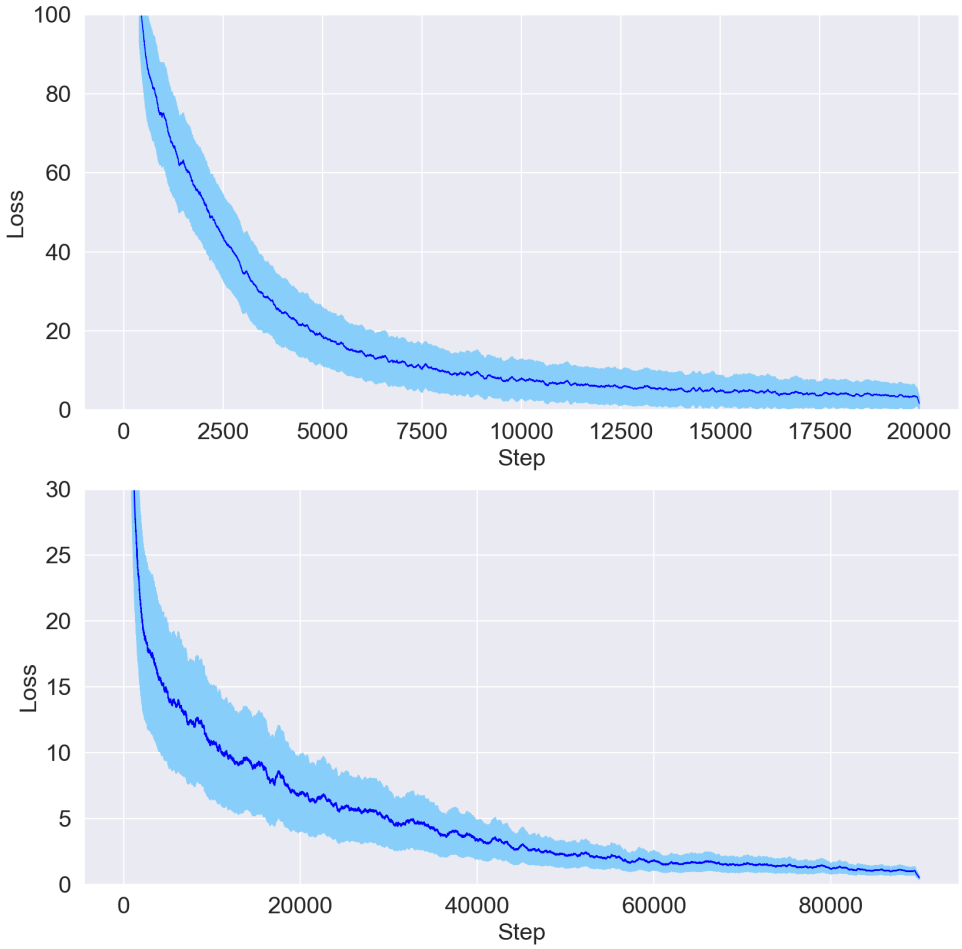

Figure 6.2: Loss versus steps for IL-based policy training under RF 1 (Top) and RF 5 (Bottom). Results are averaged over five trials with the light-blue shaded areas denoting the variance.

policies have discrete action spaces. A continuous action space may lead to unrealistic results because the trained policy may decide to take actions frequently, applying a small amount of fertilizer and irrigation every day, which is impractical for farmers to follow. Thus, we evaluated the trained policies by rounding the outputs of the network to the closest value in the action space $\mathcal{A}$ (5.1) used in RL training. The evaluation results of the trained policies are shown in Table 6.1. The application history following Trained Policy 5 and its corresponding IL-trained policy is also visualized in Figure 6.3 for illustration purpose. Based on these results, the IL-trained policies under partial observation achieved much higher cumulative rewards compared to the baseline policy with both RFs 1 and 5. In addition, the actions determined by the IL-trained policies are almost identical to the actions determined by the RL-trained policies.

Table 6.1: Performance comparison between RL-trained policies and their corresponding IL-trained policies. N_l represents the N leaching amount. Using our framework, policies under partial observation can be trained with IL and outperform the baseline policy.

	N Input (kg/ha)	Water Input (L/m^2)	N_l (kg/ha)	Yield (kg/ha)	RF 1	RF 5
Baseline Policy	360	393.7	212.6	10771.5	984.4	337.6
RL-Trained Policy 1 (Full)	200	120	35.5	10852.4	1424.7	N/A
IL-Trained Policy 1 (Partial)	200	138	37	10870.0	1407.7	N/A
RL-Trained Policy 5 (Full)	200	138	39.2	10926.1	N/A	1651.0
IL-Trained Policy 5 (Partial)	200	144	40.8	10783.7	N/A	1608.6

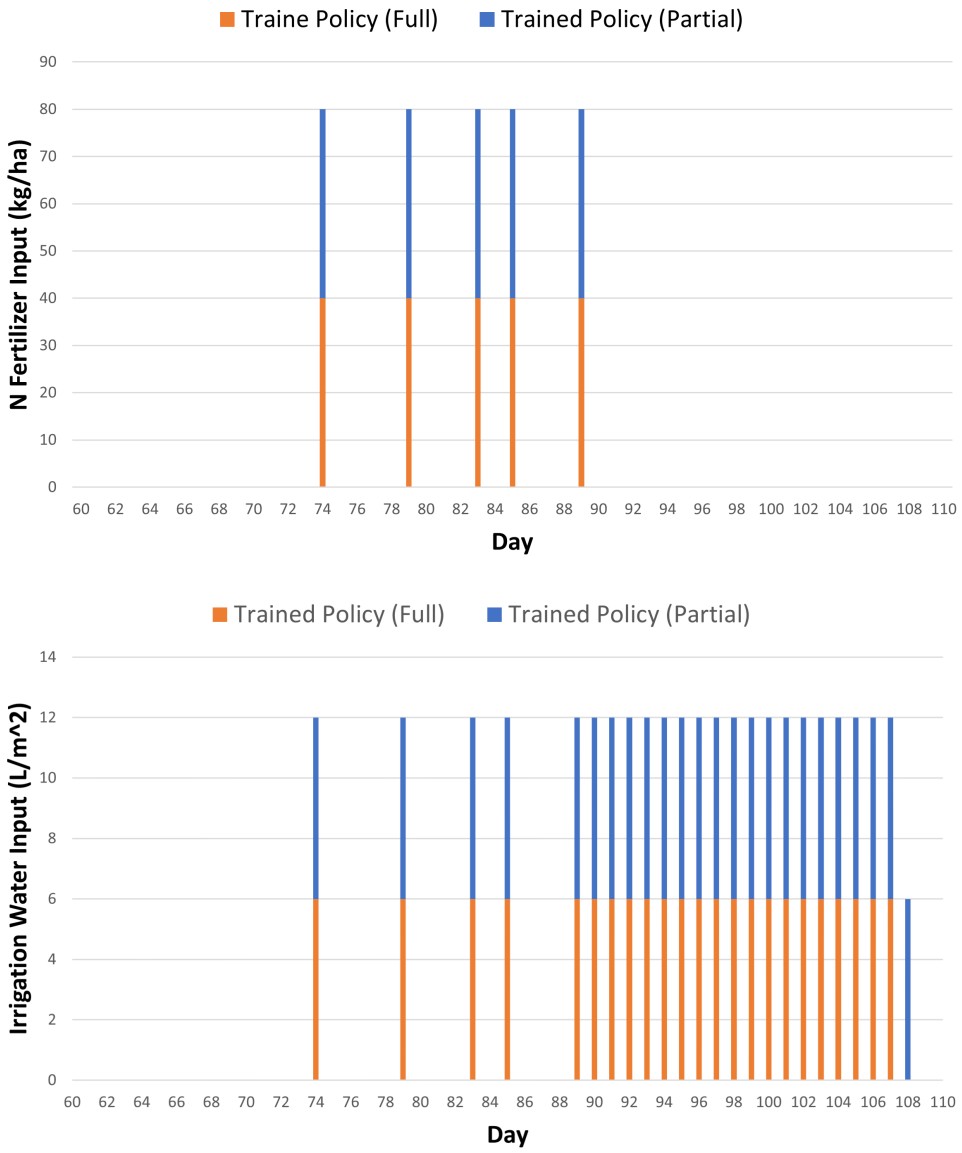

Figure 6.3: Application history of N fertilization (Top) and irrigation (Bottom) from RL-Trained Policy 5 (full) and the corresponding IL-Trained Policy 5 (partial). Both plots show values in stacked columns. The IL-trained policies under partial observation have extremely similar decisions compared with RL-trained policies under full observation.

Chapter 7

Path to Deployment

Within the RL-based training, all policies are trained using the simulated environment provided by DSSAT. Within the IL-based training, all policies are trained to mimic the actions determined from the RL-trained policies. Thus, it is very likely that all the trained management policies from the proposed framework, which work well in the DSSAT-simulated environment, may perform poorly in the real world, due to the uncertainty with weather and the mismatch between the crop models used to train the policies and the real cropping system. This is the well-known *sim-to-real gap* associated with transferring RL policies trained in simulators to the real world.

7.1 Closing the Sim-To-Real Gap

To improve the robustness of the trained management policies, we will leverage *domain and dynamics randomization* techniques, proposed to mitigate the sim-to-real gap [40], [41]. Instead of training the agent in a fixed environment, domain and dynamics randomization creates a diverse set of environments that the RL agent is exposed to during the training process, and thus the trained agent has the ability to work well with different scenarios. More specifically, we will perturb selected key parameters of the model, including soil water content, growing degree days, and vegetative growth stage, and randomize weather conditions when training the policy, which could "force" the trained policies to be robust against weather uncertainties and the mismatch between the simulation and the real world. As this thesis focuses on presenting the framework of RL and IL on crop management and validating its performance, we do not include experiments on these ideas yet, and we plan to have follow-up study addressing the robustness issue as a next step.

7.2 Deployment on a Real Farm

The IL-trained management policies under partial observation can be readily deployed on a real farm as all the 12 state variables used by the policies are easily observed or measured with sensors. Before the deployment, a farm needs to be specified for the field tests. Then, current soil and climate data corresponding to the farm need to be collected to configure the crop model within DSSAT, and shall be used for the RL-based policy training. After training, on each day, farmers or experimenters collect the current soil, crop, and weather information, and feed them into the trained policies, which generate the optimal management decisions, i.e., how much N fertilizer and water to add. Finally, farmers or experimenters can then follow these decisions to

apply the management practices. After harvest, based on the amount of water and fertilizer used and the crop yield, the performance of the management policies can be evaluated and compared with the performance from the simulation in DSSAT. Our next step is to conduct field experiments to verify the efficacy of the proposed framework.

Chapter 8

Conclusion

8.1 Summary

Finding the optimal crop management policy for N fertilization and irrigation is vital to achieving maximum yield while minimizing the management cost and environmental impact. In this thesis, we present a framework for finding the optimal management policy with deep reinforcement learning (RL), imitation learning (IL), and crop simulations based on DSSAT. Experiments are conducted for optimizing N fertilization of the maize crop in Iowa, where irrigation is not required, and optimizing N fertilization and irrigation simultaneously of maize crop in Florida, where both fertilization and irrigation are necessary for crop growth. Under full observation, i.e. with access to all variables from the simulator, deep Q-network (DQN) and soft actor-critic (SAC), two deep RL algorithms, are used to train management policies. Under partial observation, i.e., with access to a limited number of state variables from DSSAT that are observable or measurable in the real world, imitation learning is used to train management policies by mimicking the behaviors of the RL-trained policies under full observation. Given variations in potential reward functions, the trained policies from RL have different strategies during the decision-making to achieve maximum rewards. This shows that our proposed framework adapts to different scenarios. Also, using IL, the trained policies under partial observation have almost identical decisions compared to RL-trained policies under full observation. All trained policies under both full and partial observations in both Iowa and Florida achieve better results compared to two baseline methods following recommended maize production guidelines in these two locations. Furthermore, the trained policies under partial observation pave the way for real-world deployment of our framework as they only need readily accessible information.

8.2 Future Work

As we already discussed in this thesis, in the future, we plan to apply domain and dynamics randomization during the RL-based policy training to make the trained policies robust and perform well in the case of sim-to-real gap. Since the IL-trained policies under partial observation are readily deployable, we are also planning to test the efficacy of the proposed management system on a real farm by following the instructions from Section 7.2. Regarding the RL-based training, one potential improvement could be including other accessible environment information like the weather forecast to the current state variables, as we believe it may help with finding the optimal irrigation strategy. Last but not least, from an economic and practical

perspective, farmers prefer to make decisions less frequently, e.g., weekly and biweekly, instead of daily. Therefore, we plan to include different action frequencies during the RL-based training, i.e. let the agent make decisions once every week, and see how different action frequencies influence the performance of the trained policies.

References

[1] T. Searchinger, R. Waite, C. Hanson, *et al.*, *Creating a sustainable food future: A menu of solutions to feed nearly 10 billion people by 2050. final report*, 2019.

[2] B. Reddy, P. S. Reddy, F. Bidinger, and M. Blümmel, "Crop management factors influencing yield and quality of crop residues," *Field Crops Research*, vol. 84, no. 1-2, pp. 57–77, 2003.

[3] I. Ara, L. Turner, M. T. Harrison, M. Monjardino, P. DeVoil, and D. Rodriguez, "Application, adoption and opportunities for improving decision support systems in irrigated agriculture: A review," *Agricultural Water Management*, vol. 257, p. 107 161, 2021.

[4] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of artificial intelligence research*, vol. 4, pp. 237–285, 1996.

[5] A. G. Barto, P. S. Thomas, and R. S. Sutton, "Some recent applications of reinforcement learning," in *Proceedings of the Eighteenth Yale Workshop on Adaptive and Learning Systems*, 2017.

[6] V. Mnih, K. Kavukcuoglu, D. Silver, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[7] O. Vinyals, I. Babuschkin, W. M. Czarnecki, *et al.*, "Grandmaster level in StarCraft II using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[8] C. Gamble and J. Gao, *Safety-first AI for autonomous data centre cooling and industrial control*. [Online]. Available: https://deepmind.com/blog/article/safety-first-ai-autonomous-data-centre-cooling-and-industrial-control.

[9] J. Jin, C. Song, H. Li, K. Gai, J. Wang, and W. Zhang, "Real-time bidding with multi-agent reinforcement learning in display advertising," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018, pp. 2193–2201.

[10] E. Kaufmann, A. Loquercio, R. Ranftl, A. Dosovitskiy, V. Koltun, and D. Scaramuzza, "Deep drone racing: Learning agile flight in dynamic environments," in *Conference on Robot Learning*, 2018, pp. 133–145.

[11] J. Hwangbo, J. Lee, A. Dosovitskiy, *et al.*, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, 2019.

[12] Y. Song, M. Steinweg, E. Kaufmann, and D. Scaramuzza, "Autonomous drone racing with deep reinforcement learning," *arXiv preprint arXiv:2103.08624*, 2021.

[13] I. Akkaya, M. Andrychowicz, M. Chociej, *et al.*, "Solving rubik's cube with a robot hand," *arXiv preprint arXiv:1910.07113*, 2019.

[14] M. W. Palmer, J. Cooper, C. Tétard-Jones, *et al.*, "The influence of organic and conventional fertilisation and crop protection practices, preceding crop, harvest year and weather conditions on yield and quality of potato (Solanum tuberosum) in a long-term management trial," *European Journal of Agronomy*, vol. 49, pp. 83–92, 2013.

[15] J. W. Jones, G. Hoogenboom, C. H. Porter, *et al.*, "The DSSAT cropping system model," *European Journal of Agronomy*, vol. 18, no. 3-4, pp. 235–265, 2003.

[16] G. Hoogenboom, K. B. C.H. Porter, V. Shelia, *et al.*, "The DSSAT crop modeling ecosystem," in *Advances in Crop Modeling for a Sustainable Agriculture*, K. Boote, Ed., Cambridge, United Kingdom: Burleigh Dodds Science Publishing, 2019, pp. 173–216.

[17] R. Gautron, E. J. P. Gonzalez, P. Preux, J. Bigot, O.-A. Maillard, and D. Emukpere, "Gym-dssat: A crop model turned into a reinforcement learning environment," Ph.D. dissertation, Inria Lille, 2022.

[18] F. Garcia, "Use of reinforcement learning and simulation to optimize wheat crop technical management," in *Proceedings of the International Congress on Modelling and Simulation*, 1999, pp. 801–806.

[19] L. Sun, Y. Yang, J. Hu, D. Porter, T. Marek, and C. Hillyer, "Reinforcement learning control for water-efficient agricultural irrigation," in *2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications (ISPA/IUCC)*, 2017, pp. 1334–1341.

[20] H. Overweg, H. N. Berghuijs, and I. N. Athanasiadis, "Cropgym: A reinforcement learning environment for crop management," *arXiv preprint arXiv:2104.04326*, 2021.

[21] C. Ashcraft and K. Karra, "Machine learning aided crop yield optimization," *arXiv preprint arXiv:2111.00963*, 2021.

[22] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[23] J. W. Jones, J. M. Antle, B. Basso, *et al.*, "Brief history of agricultural systems modeling," *Agricultural systems*, vol. 155, pp. 240–254, 2017.

[24] T. J. Salo, T. Palosuo, K. C. Kersebaum, *et al.*, *The Journal of Agricultural Science*, vol. 154, no. 7, pp. 1218–1240, 2016.

[25] G. Brockman, V. Cheung, L. Pettersson, *et al.*, "OpenAI Gym," *arXiv preprint arXiv:1606.01540*, 2016.

[26] M. Shibu, P. Leffelaar, H. Van Keulen, and P. Aggarwal, "LINTUL3, a simulation model for nitrogen-limited situations: Application to rice," *European Journal of Agronomy*, vol. 32, no. 4, pp. 255–271, 2010.

[27] C. Zhao, B. Liu, L. Xiao, *et al.*, "A SIMPLE crop model," *European Journal of Agronomy*, vol. 104, pp. 97–106, 2019.

[28] J. Zhang and K. Cho, "Query-efficient imitation learning for end-to-end autonomous driving," *arXiv preprint arXiv:1605.06450*, 2016.

[29] T. Zhang, G. Kahn, S. Levine, and P. Abbeel, "Learning deep control policies for autonomous aerial vehicles with mpc-guided policy search," in *2016 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2016, pp. 528–535.

[30] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*, PMLR, 2018, pp. 1861–1870.

[31] C. Richardson, "Weather simulation for crop management models," *Transactions of the ASAE*, vol. 28, no. 5, pp. 1602–1606, 1985.

[32] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, J. Peters, *et al.*, "An algorithmic perspective on imitation learning," *Foundations and Trends® in Robotics*, vol. 7, no. 1-2, pp. 1–179, 2018.

[33] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Computing Surveys (CSUR)*, vol. 50, no. 2, pp. 1–35, 2017.

[34] G. Mandrini, C. M. Pittelkow, S. V. Archontoulis, T. Mieno, and N. F. Martin, "Understanding differences between static and dynamic nitrogen fertilizer tools using simulation modeling," *Agricultural Systems*, vol. 194, p. 103 275, 2021.

[35] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[36] D. Wright, J. Marois, J. Rich, and R. Sprenkel, "Field corn production guide: Ss agr 85/ag202, 1/2004," *EDIS*, vol. 2004, no. 1, 2004.

[37] G. Mandrini, C. M. Pittelkow, S. Archontoulis, D. Kanter, and N. 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*, vol. 13, 2022.

[38] C. Huang, S. W. Duiker, L. Deng, C. Fang, and W. Zeng, "Influence of precipitation on maize yield in the eastern united states," *Sustainability*, vol. 7, no. 5, pp. 5996–6010, 2015.

[39] E. Muskardin, M. Tappler, B. K. Aichernig, and I. Pill, "Reinforcement learning under partial observability guided by learned environment models," *arXiv preprint arXiv:2206.11708*, 2022.

[40] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 23–30.

[41] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 3803–3810.

Appendix A

State Variable Description

There are 28 state variables in total from the simulator and the detailed description can be found in Table A.1. Each variable usually contains one single value, except "sw", volumetric soil water content in soil layers (cm3 [water] / cm3 [soil]), which may contain multiple values depending on the number of soil layers for the given template. Most of the variables cannot be achieved by farmers in real life as they are not measurable like daily nitrogen denitrification and daily nitrate leaching. For the state variables used in the partial observation study, all of them can be easily accessed or measured by farmers in the real world, like the weather information, soil water content, which can be measured by a soil moisture meter, and maize growing stage, which can be easily identified by an experienced farmer.

Table A.1: State variable description

Variable	Description	Included in Partial Observation Study?
cumsumfert	cumulative nitrogen fertilizer applications (kg/ha)	Yes
dap	days after simulation started	Yes
dtt	growing degree days for current day (C/d)	Yes
istage	DSSAT maize growing stage	Yes
vstage	vegetative growth stage (number of leaves)	Yes
pltpop	plant population density (plant/m2)	Yes
rain	rainfalls for the current day (mm/d)	Yes
srad	solar radiations during the current day (MJ/m2/d)	Yes
tmax	maximum temparature for current day (C)	Yes
tmin	minimum temparature for current day (C)	Yes
sw	volumetric soil water content in soil layers (cm3 [water] / cm3 [soil])	Yes
xlai	plant population leaf area index (m2_leaf/m2_soil)	Yes
nstres	index of plant nitrogen stress (unitless)	No
pcngrn	massic fraction of nitrogen in grains (unitless)	No
swfac	index of plant water stress (unitless)	No
tleachd	daily nitrate leaching (kg/ha)	No
grnwt	grain weight dry matter (kg/ha)	No
cleach	cumulative nitrate leaching (kg/ha)	No
cnox	cumulative nitrogen denitrification (kg/ha)	No
tnoxd	daily nitrogen denitrification (kg/ha)	No
trnu	daily nitrogen plant population uptake (kg/ha)	No
wtnup	cumulative plant population nitrogen uptake (kg/ha)	No
topwt	top weight (kg/ha)	No
es	actual soil evaporation rate (mm/d)	No
runoff	calculated runoff (mm/d)	No
wtdep	depth to water table (cm)	No
rtdep	root depth (cm)	No
totaml	cumulative ammonia volatilization (kgN/ha)	No