

Transformer Infused Deep Reinforcement Learning for Resource Allocation in Crop Production

Daniel RuiRu^{1,2}[0000–0003–3197–8821], Nicolas Jouandeau²[0000–0001–6902–4324], and
Dickson Owuor¹[0000–0002–0968–5742]

¹ SCES, Strathmore University, Nairobi, Kenya

druiru@strathmore.edu & dowaor@strathmore.edu

² LIASD, Université Paris 8, Paris, France

nicolas.jouandeau@univ-paris8.fr & daniel.kiguru@etud.univ-paris8.fr

Abstract. The agricultural sector faces serious challenges in meeting the needs of food security. Crop management is one of these challenges, where a balance between productivity and environment impact is needed. Such a complex issue is seen in fertilizer application as crop management practices try to increase crop yields while reducing environmental impacts. This issue represents a classical case of resource allocation problem where machine learning can be applied. Predictive methods like Deep Reinforcement Learning (DRL) are gaining popularity in solving such problems, which this paper uses to sustainably apply fertilizers to crops. This paper presents a fairly new solution to this problem by combining transformers with DRL agents. These transformer-based DRL agents are trained in the same environment as baseline DRL agents to compare their performance. Transformer-based agents show considerable superiority in balancing the needs of crop production and environmental sustainability. Ultimately, transformer-based agents get better yields and cumulative reward without applying excessive amounts of nitrogen fertilizer.

Keywords: Deep Reinforcement Learning · Deep Q Network · Proximal Policy Optimization · Transformer · CropGym.

1 Introduction

According to the Sustainable Development Goals (SDGs), the agricultural sector faces serious challenges in addressing food security. Current crop production methods are still not adequate to sustain the world’s population and continue to face increasingly high demands. Recent projections show the severity of these demands, as human population is expected to reach 9.6 billion by the year 2050 [1]. Moreover, the agricultural industry will have to reduce environmental impacts while meeting the demands of food security [2]. Various factors of crop production will influence the results of these expectations. According to [3] the use of nitrogen fertilizer and irrigation are the most significant factors as they heavily influence crop management and results.

Current empirical studies are heavily focused on the best practices of applying nitrogen fertilizer and irrigation without considering their optimality. There are even more negligible studies that show whether these existing strategies can perform well under

changing market and climate conditions. It is therefore important to have new methods to tackle the needs of crop management while addressing the sustainability of administering nitrogen fertilizer and the irrigation of crops [4]. The new agricultural strategies should balance the factors of climate, yield prices and cost management. Deep Reinforcement Learning (DRL), a combination of Reinforcement Learning (RL) and Deep Learning (DL) has demonstrated exceptional ability in solving complex sequential decision making problems. Problems like those in robotics have been solved using DRL, results that can be replicated in crop production [5]. Representing a classic case of resource allocation problem, crop management can be treated as a typical Markov Decision Process (MDP)¹ in which DRL can be deployed.

DRL can provide efficient crop production processes by enhancing productivity and mitigating negative environmental impacts. This efficiency can be achieved by optimizing practices of crop production [6]. Crucial to this benefit is the timing and sequencing of these practices, which must align with the needs of crop types, sowing methods, and harvesting. Variables like soil conditions, plant age and uncertainties of climate change should also be considered as constraints. The existing crop management methods can hardly keep up with all these requirements, which explains the recent push by academia and industry in finding alternative solutions [7].

The experiments of this paper led to one scientific contribution; the development of transformer-based DRL agents for crop management practices. These agents were based on a representation learning formulation that is defined in the methodology. Using the transformer architecture helped to dynamically weigh the importance of the learning environments, improving the final rewards.

2 Literature Review

2.1 Related Works

DRL in Resource Allocation: Resource allocation (RA) problems are ubiquitous with most modern processes. From financial services to agriculture and even home activities. There is always a need to schedule tasks, as a basic level of RA [8]. Existing allocation methods use heuristics, where rule-based techniques decide how resources are distributed. As observed by [8], most resource allocation solutions are largely designed using a two-step process; (1) developing a clever heuristic that defines a simple model of the problem and (2) diligently testing and fine tuning the heuristics to get good results. This process is repeated many times more so, when certain aspects of the problem change, which is extremely inefficient.

To improve these rule-based methods, a number of predictive solutions have been suggested. Revolving around the use of machine learning, several resource allocation problems have been studied with a wide range of algorithms developed to solve them. Specific to the DRL domain, most scholars have attempted to develop solutions that adapt to changes in specific environments. For example [9] and [10] attempted to solve allocation problems in business processes using predictive models where the former

¹ An MDP is the mathematical formulation that models sequential decision-making. It defines states, actions and rewards.

used offline prediction models based on LSTM while the latter used double DRL to get optimal resource allocation policies. [8] tested DRL on computing resources as they tried to develop an optimal scheduling algorithm that in the end showed great promise in its domain, including outperforming existing ad-hoc heuristic methods. [11] presented another notable contribution, as they improved existing Deep Q-Networks (DQN) to develop a resource allocation policy for edge applications in the Internet of Things (IoT).

Although these examples of DRL in resource allocations are not exhaustive, they showcase the impact of DRL in these spaces. They also paint a bigger picture of why DRL and other predictive models are becoming common in resource allocation problems. Mainly, this drive is led by the nature of the problems that are characterized by constantly changing conditions [12].

DRL in Agriculture: Agriculture and its demands for resource allocation has seen DRL deployed in different capacities. [13] used DRL in a general capacity to optimize agricultural practices while minimizing environmental impact. Their work used Proximal Policy Optimization (PPO) to improve fertilizer and irrigation management practices. A similar process was followed by [14] who applied PPO in crop management to achieve higher yearly profits (11 percent in a year) as compared to other expert methods. [15] applied DRL in crop management for the greater objective of Decision Support System for Agrotechnology Transfer (DSSAT) [16]. DSSAT is a widely used crop model similar to CropGym that is used in this paper [17].

Closer to the experiments of this paper, several studies have used DRL/RL for crop management. However, the study by [18] stands out because of two reasons; one, they developed the CropGym environment. Secondly, their study successfully used PPO to learn fertilization policies that also minimized environmental impacts. Like other studies in this domain of agriculture, a plan to logically combine crop production practices was used as the underlying concepts of the CropGym environment. Additionally, the environment and policies developed served as the backbone of agricultural decision support systems (DSS), where the crop processes were optimized [19]. These related works of DRL in agriculture highlight resource management processes where models describe growth, yields, and environmental conditions. Table 1 lists some notable studies in this domain and their limitations.

Table 1: Related works on DRL in agriculture and their limitations

Related Work & Description	Limitations
[13] RL in agriculture	Limited scalability; lacks field trials
[14] PPO in multiple crops	Scalability issues; limited discussion of data requirements
[15] DRL in crop management	Only nitrogen fertilizer optimized
[16] Introduced DSSAT	Data challenges; simplified irrigation models
[18] Introduced CropGym	Scalability issues
[20] RL environment for irrigation	High environment complexity
[21] DQN in irrigation scheduling	Risk of overfitting; limited generalization
[22] RL and digital twin in agriculture	Limited real-world applications; data integration challenges
[23] DRL in irrigation scheduling	Performance variability
[24] RL and LLM in crop production	Integration complexity; scalability and adaptability issues

2.2 Transformer Based DRL

Recent developments in neural networks have seen several scholars try to integrate Transformers with DRL. Transformers are a group of neural network architectures that consist of multiple layers, each having multi-head self attention mechanisms [25]. These architectures also have parallel fully-connected networks, residual connections [26], and layer normalization [27]. When Transformers are given a sequence of input embeddings n , they produce an n output embeddings, each representing the relationship between the inputs and the rest of the input sequences. Although the original examples of this application was in Natural Language Processing (NLP), transformers can also be applied in other settings with different inputs. For instance, the inputs can be state spaces from a DRL environment.

In this DRL application, the self attention mechanism of transformers enables input embeddings (rows of X) to attend to every other embedding in an input sequence. This ability is facilitated by the computation of attention scores that is done using Query², Key³ and Value⁴ tensors [25]. Fig. 1 summarizes the attention mechanism of Transformers. More information on this process and Transformers can be found in the original paper [25] and other follow-up publications. The paper by [28] describes the self-attention mechanism and can be supplemented by [26] that describes residual connections.

² Represents current token looking for meaning in a sequence, given as $Q = XW_q$.

³ An index tag for every token & matches against query to score relevance, given as $K = XW_k$.

⁴ Semantic meaning of a token, given as $V = XW_v$.

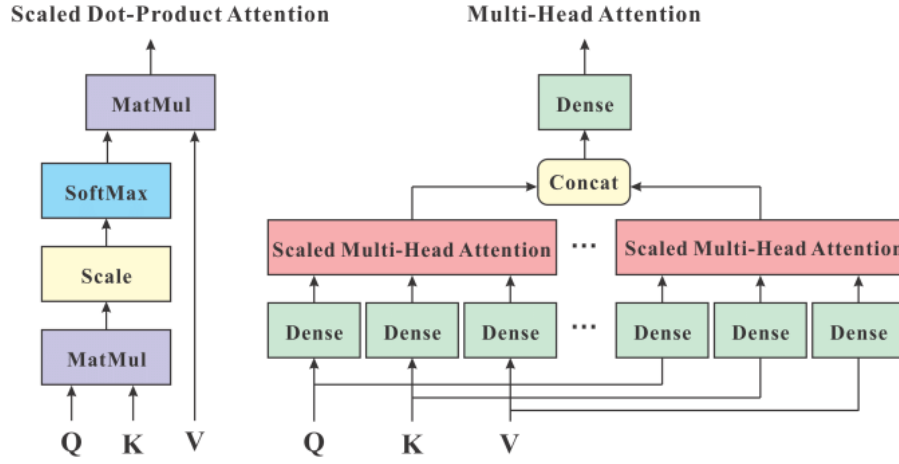


Fig. 1: Scaled dot product attention and multi-head attention [29]

Several notable publications have combined transformers with RL for resource allocation.

- [30] achieved good results in generalizable resource allocation in stream processing by using LSTM-based encoder-decoder models. Their work got state-of-the-art results in most of their test cases.
- [31] introduced Decision Transformer (DT) which redefined RL as a sequence problem. DT achieved state-of-the-art results in games like Atari, Key-to-Door tasks and OpenAI Gym environments.
- [32] employed a transformer based RL to dispatch dynamic job shop scheduling. Their model outperformed traditional dispatching methods, generic algorithms and even classical RL solutions.
- [33] developed an innovative DRL framework using transformers to schedule tasks in parallel machine scheduling problem (PMSP).
- [34] combined transformers with neural networks to balance user satisfaction needs and fairness of Point of Interest (POI) recommendations. Their model outperformed baseline solutions in both user satisfaction and POI-based evaluation metrics.

Variations of Transformer Based DRL: These studies that combine transformers and RL/DRL try to leverage the power of the former in stochastic environments. Transformers in RL can provide significant advantages including sequence modeling of states to help RL agent focus on specific cues of environments [35]. In complex environments, transformers can help with handling different data modalities [36], and facilitate parallel processing which leads to efficient run times for training policies [37]. Finally, scalability, a key struggle of complex tasks, can be improved by transformers because they have smooth improvements [38]. However, the combination of transformers and RL is

not without its challenges starting with the sensitivity of RL algorithms to deep neural networks that heavily affects performance [39]. Secondly, there is the complexity of the resultant models that often have high latency and large memory footprints [40]. These challenges can be addressed by finding efficient combinations, which can be grouped into the following categories.

- **Representation learning** - Meaningful representations determine the decision-making ability of RL agents [41]. Empirical studies show that training RL agent on high dimensional data e.g. images causes sample inefficiency. Good data representation determines the learning done by RL policies [42] as it improves performance, policy stability and convergence speed [43]. Transformers meet this role of representation by providing descriptive feature representation for different data modalities [44].
- **Transition function learning** - The transition function in RL describes the environments’ transitions from current state to the next. Transformers can be employed to the dynamics of a simulator environment by modeling the transition function. This modeling is achieved by leveraging the collected trajectories of environments [45]. A transformer-based RL model can then train on these trajectories.
- **Reward learning** - Learning the reward function can be improved by transformers. This component of RL quantifies the desirability of actions in a given state. Recent studies have shown that learning reward function by integrating human data⁵ gives better rewards [27]. This integration can benefit from Transformers as these neural networks have demonstrated superiority in dealing with human data.
- **Policy learning** - Learning policies can also be improved using Transformers in both offline and online settings. For offline RL, a policy is trained using static datasets from collected experiences. In such scenarios, combinations like those of Decision Transformers (DT) [31] can predict actions that give the best returns. In online RL, Transformers can provide a solution to issues of noisy sensor, unknown agents and problems of partial observability [46].

3 Methodology

3.1 Experiments Setup

The Environment - CropGym: CropGym is a Gymnasium/Gym environment for simulating farm management policies. This environment test farm processes like fertilization and irrigation by using a process-based crop growth model [19]. CropGym was built on top of the Python Crop Simulation Environment (PCSE), an open source framework that has implemented a variety of crop simulation models such as LINTUL-3 (Light Interception and Utilization) [47] and WOFOST (World Food Studies) [48]. CropGym is highly configurable with various adjustment available for input parameters, like crop attributes and driving variable such as weather information. It also has functions to facilitate observational data such as field measurements.

⁵ For example real time feedback and expert demonstrations (past examples, plays etc.).

CropGym follows standard RL design as defined by the gym conventions. The underlying code is modular which enables users to create custom environments [19]. The standard environment has the following components:

State space (S) - made of current state of crop and weather observations. Farmers have a natural intervention of a week and this explains the default weekly setting for observations. Observations by the agent consist of outputs from the LINTUL-3 model. This model is designed to simulate winter weather [49]. Table 2 shows some of the observed variables in CropGym.

Table 2: CropGym - Crop growth and weather variable

Environment Variable	Meaning
WSO	Weight of Storage Organ
TNSOIL	Amount of nitrogen in the soil
LAI	Leaf area index
NUPTT	Total N uptake
DVS	Development stage
TRAIN	Total rainfall
WC	Soil moisture content

Action space (A) - consists of discrete doses of nitrogen fertilizer. Three application amounts are specified; 0, 20 and 40 kg/ha. **Transitions between states** - This is governed by a deterministic process-based model (LINTUL-3) and weather sequences [47]. Weather sequences are based on data obtained from PowerNASA database and the locations specified using the PowerNASAWeatherDataSource API for the years 1990 to 2022 [50]. The current experiments used weather data from Ol’ Kalou. **Reward function (R)** - designed to give higher yields with limited fertilizer usage. This reward tries to balance the gains in production and environmental costs (constraint). R is formalized as:

$$R_t = (WSO_t^\pi - WSO_{t-1}^\pi) - (WSO_t^\theta - WSO_{t-1}^\theta) - \beta N_t \quad (1)$$

Where:

- t is the timestep
- WSO is the weight of storage grain (g/m²) and
- N is the nitrogen amount (g/m²)

The upper indices notations π and θ represent an agent’s policy and zero nitrogen policy [19]. An added parameter β defines the trade-off between higher yields and reduced environmental impact. For example, setting β to 2.0 reflects a reward whose objective is economic profitability because a kilogram of fertilizer is as twice as expensive as a kilogram of wheat [19]. This paper used β value of 10.0 (focus on environmental impact).

Baseline Agents: Standard DQN and PPO algorithms were used as the baseline agents. These standard agents applied nitrogen fertilizer on three times within a production season (i.e. 3rd March, 31st March and 5th May). Both agents are based on the original definitions done by [51] for DQN and [52] for PPO. The former model was implemented using convolutional neural networks trained on a variant of Q-learning which led to a deep learning model for reinforcement learning. PPO is a policy gradient method for reinforcement learning. It is defined to alternate between sampling data from environment interactions and optimizing a surrogate objective function.

Like the original study by [18], these baseline agents were implemented using the Stable Baselines3 packages and PyTorch [53]. The DQN agent is defined by a policy network with two fully connected layers of 256 units each. The PPO agent has both policy and value function networks, both having two fully connected layers and 256 units. Table 3 gives a summary of the default hyper-parameters.

Table 3: Baseline agents hyper-parameters

Hyper-parameter	Value
Batch size	64
No. steps to run/update	2048
Learning rate	0.0003
Clip range	0.3
No. epochs	10
GAE lambda	0.95
Value function coefficient	0.5
Entropy Coefficient	0.0
Max gradient norm	0.5
Exploration fraction	0.1
Exploration initial epsilon	1.0
Exploration final epsilon	0.01
Activation function	nn.Tanh

Proposed Models: Transformer-Based DRL Algorithms: Both baseline DQN and PPO were modified by adding transformer networks to improve representation learning. As seen in sub-section 2.2, this application is meant to provide an agent with meaningful representation of an environment for better decision making.

For transformer-based DQN (Tx-DQN), a TransformerFeatureExtractor class was defined to process observations before they were feed into the Q network. This class replaced a standard feedforward encoder that is usually the feature extractor. A transformer encoder mapped raw input observations into a better latent feature representation. This new encoder also had a learnable and dynamic positional encoding which helped with non-fixed input structures as shown in Fig. 2. Each transformer encoder layer had:

- **Multi-head Self-Attention layer** - using scaled dot product attention.

- **Add and Norm**⁶ - to preserve gradients and normalize outputs
- **Position-wise feedforward network** - two fully connected layers and activations between them.
- **Dropout layers** - used after attention and feedforward network.

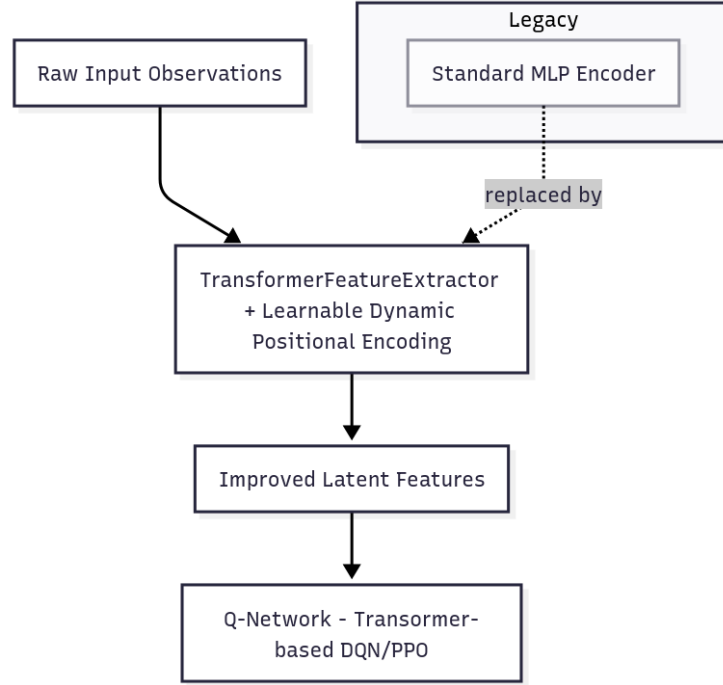


Fig. 2: Transformer-based DRL algorithms

A similar application of transformer was used with the transformer-based PPO (Tx-PPO). TransformerFeatureExtractor was added as a custom feature extractor for observations which aimed to give a richer and contextualized embeddings. This encoder also had a learnable and dynamic positional encoding. Like the Tx-DQN, Tx-PPO had self attention and feedforward networks. The embeddings from this custom feature extractor were fed into PPO's policy and value function networks separately.

3.2 Training the Models

The training process compared the baseline agents and Transformer Based DQN/PPO agents. Tables 4 and 5 show the hyperparameters of the transformer based agents. Subsequently, the generalization of these agents was tested using a different testing location, other than the training location.

⁶ Includes residual connections plus layer normalization.

Table 4: Transformer-based DQN agent hyper-parameters

Hyper-parameter	Value	Description
Feature dimension (F)	256	Size of transformer I/O
No. heads	4	No. of attention heads
No. layers	4	No. of stacked transformer encoder layers
Dropout	0.2	Dropout rate
Positional encoding	learnable(size [1, 50, F])	Positional encoding
Activation	ReLU	The activation function
Learning rate	1e-5	A linear schedule
Buffer size	100000	Size of replay buffer
Learning start	10000	Steps before learning begins
Batch size	64	Minibatch (replay buffer)
Tau	0.005	Target network update coefficient
Gamma	0.99	Discount factor
Exploration fraction	0.2	Fraction for training
Exploration initial epsilon	1.0	Initial exploration rate
Exploration final epsilon	0.01	Final epsilon value
Target update interval	2000	Frequency for target net update
Train frequency	4	Train after every 4 steps
Gradient steps	4	Gradient steps per update

Training the Agents: The training was done on a personal computer (PC) with 11th generation Intel core i7-11850H x 16 specifications. The PC has a NVIDIA T600 GPU, 32 gigabyte of RAM and 1 terabyte of storage space. Each experiment had an average training time of 43,200 seconds for the baseline agents and 45,000 seconds for the transformer-based agents.

Three variables were used to evaluate the agents.

- **Cumulative reward** - highlighted the total reward an agent achieved over the evaluation episodes. The cumulative reward was derived from WSO (Weight Storage Grain) as illustrated in subsection 3.1
- **Weight Storage Grain (WSO)** - a key output variable that represents the harvestable yield. WSO is calculated during the reproductive phase using the assimilated partition of the total biomass⁷. WSO in the experiments evaluated the effectiveness of the agents' decisions (in applying fertilizer) to maximize yields.
- **Total soil inorganic nitrogen (TNSOIL)** - a representation of the total amount of plant-available nitrogen in the soil, specifically in the form of inorganic nitrogen. TNSOIL serves as an observation feature that also affects the reward function by penalizing excessive nitrogen use. This constraint encourages sustainable crop management.

Running the Experiments: 100 different experiments were conducted using both Optuna and manual hyper parameter adjustments. Table 6 and 7 shows some of the manual

⁷ Grain for this study because the crop used was wheat.

Table 5: Transformer-based PPO agent hyper-parameters

Hyper-parameter	Value	Description
Feature dimension (F)	256	Size of transformer I/O
No. heads	4	No. of attention heads
No. layers	4	No. of stacked Tx encoder layers
Dropout	0.2	Dropout rate
Positional encoding	learnable(size [1, 50, F])	Positional encoding
Activation	ReLU	The activation function
Input dimension	np.prod	Flattened dimension (dynamic)
Learning rate	1e-5	A linear schedule
No. steps	2048	Environment steps
Buffer size	100000	Size of replay buffer
Learning start	10000	Steps before learning begins
Batch size	64	Minibatch (replay buffer)
No. epochs	10	Optimization epochs per update
Gamma	0.99	Discount factor
Gae lambda	0.95	Generalized advantage estimation
Clip range	0.2	PPO policy update clipping rate
Clip range value function	0.2	Clipping rate for value function
Entropy Coefficient	0.01	Entropy
Value function coefficient	0.5	Coefficient for vf loss in total loss
Max gradient norm	0.5	Gradient clipping threshold

adjustments. The baseline and transformer based agents trained for 420000 steps. The agents were trained on the same location, Ol’Kalou, a region in central Kenya (Latitude -0.27, Longitude 36.39 and 2,348 - 2,380 meters above sea level). Subsequently, the testing locations were Ol’Kalou and Ishiara ⁸ (Latitude -0.53, Longitude 37.49 and 803 - 1,021 meters above sea level). This second test location was used to evaluate the agents’ generalization capabilities.

Table 6: Summary of DQN combinations

Exp	exp_fn	exp_eps0	exp_final_eps	net_arch	act_fn
1	0.1	1.0	0.01	[256, 256]	nn.Tanh
2	0.2	1.0	0.01	[256, 256]	nn.Tanh
3	0.1	0.8	0.01	[256, 256]	nn.Tanh
4	0.1	1.0	0.05	[256, 256]	nn.Tanh
5	0.1	1.0	0.01	[512, 512]	nn.Tanh
6	0.1	1.0	0.01	[128, 128]	nn.ReLU
7	0.2	1.0	0.1	[512, 512]	nn.ReLU

⁸ A region in central-eastern part of Kenya.

Table 7: Summary of PPO combinations

Exp	n_steps	l_rate	clip_range	gae_lambda	net_arch	act_fn
1	2048	0.0003	0.3	0.95	[256, 256]	nn.Tanh
2	2048	0.0001	0.3	0.95	[256, 256]	nn.Tanh
3	4096	0.0003	0.2	0.95	[512, 512]	nn.ReLU
4	2048	0.0003	0.3	0.98	[128, 128]	nn.Tanh
5	2048	0.0003	0.1	0.95	[256, 256]	nn.ReLU
6	4096	0.0005	0.2	0.95	[512, 512]	nn.Tanh
7	4096	0.0001	0.4	0.92	[128, 128]	nn.ReLU

4 Results and Discussion

4.1 WSO and Cumulative Reward

All agents were designed with an objective of finding an optimal policy that would apply nitrogen fertilizer to get good yields and reduce environmental costs. The baseline agents performed fairly well, achieving good values for WSO and cumulative reward. However, as shown in table 8, the transformer based agents outperformed the base agent in most of the evaluation episodes. By having a higher WSO, Tx-DQN and Tx-PPO were allocating growth resources (nitrogen) better to facilitate a higher harvest-able yield. This outcome is validated by Table 9 that subsequently saw Tx-DQN and Tx-PPO have a higher cumulative reward.

Recall, these higher values of cumulative reward were constrained to facilitate environmental sustainability using the β component of the reward. Therefore, the Tx-based agents achieved higher yields while still limiting the effects of artificial fertilizer in the environment.

Table 8: WSO results over training steps

Step	Base DQN	Base PPO	Tx-DQN	Tx-PPO
20000	37.48	35.48	46.78	48.31
61500	55.09	56.09	50.11	42.11
103500	64.74	65.75	75.14	69.31
145500	44.92	40.79	67.63	66.12
187500	25.69	28.11	39.36	26.99
229500	25.70	27.23	39.19	27.70
271500	41.57	40.99	60.60	49.22
313500	20.43	22.79	105.47	90.68
355500	31.54	33.23	60.36	54.58
397500	30.89	31.89	123.88	101.04

*Values shown are based on down-sampled data (averaged across 100 experiments)

Table 9: Cumulative reward over training steps

Step	Base DQN	Base PPO	Tx-DQN	Tx-PPO
15000	-23.53	-23.65	-21.75	-22.118
52500	5.78	5.77	-1.16	11.81
92500	5.77	6.65	12.48	9.31
132500	8.54	7.23	13.90	13.30
172500	5.73	7.98	12.93	15.01
212500	8.96	7.65	13.35	5.47
252500	12.03	13.45	15.73	-19.84
292500	8.12	8.12	11.83	12.85
332500	12.00	12.45	14.35	15.22
372500	12.16	12.12	14.80	15.45

*Values shown are based on down-sampled data (averaged across 100 experiments)

4.2 Ol’Kalou and Ishiara Reward

To test for generalization beyond the objective of increasing crop yields, tests were done in a region not included in the training location. The training of all the agents was done in Ol’Kalou. Testing on the same region gave high results for the cumulative reward (table 10) just as observed in table 9 through the general evaluations.

For the new test region (not in the training location), negative cumulative rewards were observed. Even the Tx-based agents failed to get any promising results, only outperforming the baseline agents by getting zero cumulative reward. These Tx-based agents also showed promise in this generalization test by having a cumulative reward that rose steadily from negative values to zero as shown in table 11.

Table 10: Ol’Kalou reward over training steps

Step	Base DQN	Base PPO	Tx-DQN	Tx-PPO
20000	43.15	41.79	76.92	55.86
61500	22.46	24.49	71.83	117.77
103500	65.90	61.16	126.49	147.53
145500	128.10	129.90	137.31	144.26
187500	25.47	24.75	134.19	116.46
229500	70.89	71.04	117.83	71.95
271500	16.59	17.46	131.36	54.20
313500	53.60	52.09	138.08	28.13
355500	58.26	59.83	120.23	33.63
397500	47.28	45.89	115.03	69.42

*Values shown are based on down-sampled data (averaged across 100 experiments)

Table 11: Ishiara reward over training steps

Step	Base DQN	Base PPO	Tx-DQN	Tx-PPO
20000	-16.34	-16.01	-15.85	-23.17
61500	0.00	0.00	-3.33	0.00
103500	-4.46	-5.71	-1.90	0.00
145500	-2.86	-3.46	-20.95	0.00
187500	-0.95	-0.56	-30.95	0.00
229500	0.00	0.00	-9.05	-14.29
271500	-4.76	-3.97	-2.86	0.00
313500	0.00	0.00	0.00	0.00
355500	-3.96	-4.76	0.00	0.00
397500	-19.05	-18.33	0.00	0.00

*Values shown are based on down-sampled data (averaged across 100 experiments)



Fig. 3: Smoothed training reward over time

4.3 Training Reward, Training Loss and Policy Gradient Loss

The better performance by the Tx-based agents was also supported by the training metrics. Fig. 3 shows the training reward and it rose steadily, to a higher values for Tx-based agents. The training loss and policy gradient also reduced faster in Tx-based agents compared to the baseline agents.

5 Conclusion and Future Work

The transformer based agent were envisioned as better alternatives in allocating resources in crop production. Their main advantage came from a better state representation that gave the RL agents richer environments to train on. This edge extended to the training processes, where the agents had better training metric results. The only drawback to their performance was their generalization to unknown regions. Although, objectively, they did better than the baseline agents. Better results could be observed with more adjustments to the transformer networks and the collaboration done with the RL agents (e.g. policy learning and not just representation). Future work could consider these experiments including the addition of human experiences to the training environment.

Acknowledgments. This work was performed using HPC resources from GENCI-IDRIS (Grant 2025-[AD010614071R2]). The researchers thank Strathmore University and LIASD Lab (Université Paris 8) for invaluable academic and institutional support during the research of this paper. The lead author gratefully acknowledges the French Embassy in Kenya for its generous support and facilitation of his doctoral studies.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. R. Tao, P. Zhao, J. Wu, et al., “Optimizing crop management with reinforcement learning and imitation learning,” *arXiv*, 2022, doi: 10.48550/arXiv.2209.09991.
2. 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.
3. J. Wu, D. Pichler, D. Marley, et al., “Extended Agriculture-Vision: An extension of a large aerial image dataset for agricultural pattern analysis,” *arXiv*, 2023, doi: 10.48550/arXiv.2303.02460.
4. I. Ara, L. Turner, M. T. Harrison, et al., “Application, adoption and opportunities for improving decision support systems in irrigated agriculture: A review,” *Agric. Water Manage.*, vol. 257, pp. 107161, 2021, doi: 10.1016/j.agwat.2021.107161.
5. E. Kaufmann, A. Loquercio, R. Ranftl, et al., “Deep drone racing: Learning agile flight in dynamic environments,” in *Proc. Conf. Robot Learn.*, pp. 133–145, 2018.
6. D. Chen and Y. Huang, “Integrating reinforcement learning and large language models for crop production process management optimization and control through a new knowledge-based deep learning paradigm,” *Comput. Electron. Agric.*, vol. 232, pp. 110028, 2025, doi: 10.1016/j.compag.2025.110028.

7. A. Skhiri and F. Dechmi, "Impact of sprinkler irrigation management on the Del Reguero river (Spain). I: Water balance and irrigation performance," *Agric. Water Manage.*, vol. 103, pp. 120–129, 2012, doi: 10.1016/j.agwat.2011.11.001.
8. H. Mao, M. Alizadeh, I. Menache, and S. Kandula, "Resource management with deep reinforcement learning," in *Proc. 15th ACM Workshop Hot Topics Netw.*, pp. 50–56, 2016, doi: 10.1145/3005745.3005750.
9. G. Park and M. Song, "Prediction-based resource allocation using LSTM and minimum cost and maximum flow algorithm," in *Proc. Int. Conf. Process Mining (ICPM)*, pp. 121–128, 2019, doi: 10.1109/ICPM.2019.00029.
10. K. Zbikowski, M. Ostapowicz, and P. Gawrysiak, "Deep reinforcement learning for resource allocation in business processes," in *Proc. Int. Conf. Process Mining*, pp. 177–189, 2022, doi: 10.1007/978-3-031-08848-3-11.
11. X. Xiong, K. Zheng, L. Lei, et al., "Resource allocation based on deep reinforcement learning in IoT edge computing," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 1133–1146, 2020, doi: 10.1109/JSAC.2020.2986624.
12. G. Zhou, W. Tian, R. Buyya, et al., "Deep reinforcement learning-based methods for resource scheduling in cloud computing: A review and future directions," *Artif. Intell. Rev.*, vol. 57, no. 5, pp. 1–42, 2024, doi: 10.1007/s10462-024-10764-3.
13. R. Gautron, O.-A. Maillard, P. Preux, et al., "Reinforcement learning for crop management support: Review, prospects and challenges," *Comput. Electron. Agric.*, vol. 200, pp. 107182, 2022, doi: 10.1016/j.compag.2022.107182.
14. M. Turchetta, L. Corinzia, S. Sussex, et al., "Learning long-term crop management strategies with CyclesGym," in *Adv. Neural Inf. Process. Syst.*, vol. 35, pp. 11396–11409, 2022.
15. J. Wu, R. Tao, P. Zhao, et al., "Optimizing nitrogen management with deep reinforcement learning and crop simulations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 1712–1720, 2022, doi: 10.1109/CVPR52688.2022.00177.
16. J. W. Jones, G. Hoogenboom, C. H. Porter, et al., "The DSSAT cropping system model," *Eur. J. Agron.*, vol. 18, no. 3–4, pp. 235–265, 2003, doi: 10.1016/S1161-0301(02)00107-7.
17. R. Gautron, E. J. Padrón, P. Preux, et al., "gym-DSSAT: A crop model turned into a reinforcement learning environment," *arXiv*, 2022, doi: 10.48550/arXiv.2207.03270.
18. H. Overweg, H. N. C. Berghuijs, and I. N. Athanasiadis, "CropGym: A reinforcement learning environment for crop management," *arXiv*, 2021, doi: 10.48550/arXiv.2104.04326.
19. M. Kallenberg, H. Overweg, R. van Bree, et al., "Nitrogen management with reinforcement learning and crop growth models," *Environ. Data Sci.*, vol. 2, pp. e34, 2023, doi: 10.1017/eds.2023.31.
20. M. Madondo, M. Azmat, K. Dipietro, et al., "A SWAT-based reinforcement learning framework for crop management," *arXiv*, 2023, doi: 10.48550/arXiv.2302.04988.
21. K. Alibabaei, P. D. Gaspar, E. Assunção, et al., "Irrigation optimization with a deep reinforcement learning model: Case study on a site in Portugal," *Agric. Water Manage.*, vol. 263, pp. 107480, 2022, doi: 10.1016/j.agwat.2022.107480.
22. G. Goldenits, K. Mallinger, S. Raubitzek, et al., "Current applications and potential future directions of reinforcement learning-based digital twins in agriculture," *Smart Agric. Technol.*, vol. 8, pp. 100512, 2024, doi: 10.1016/j.atech.2024.100512.
23. T. Kelly, T. Foster, and D. Schultz, "Assessing the value of deep reinforcement learning for irrigation scheduling," *Smart Agric. Technol.*, vol. 7, pp. 100403, 2024, doi: 10.1016/j.atech.2024.100403.
24. D. Chen and Y. Huang, "Integrating reinforcement learning and large language models for crop production process management optimization and control through a new knowledge-based deep learning paradigm," *Comput. Electron. Agric.*, vol. 232, pp. 110028, 2025, doi: 10.1016/j.compag.2025.110028.

25. A. Vaswani, N. Shazeer, N. Parmar, et al., “Attention is all you need,” in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 30, 2017.
26. K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 770–778, 2016, doi: 10.1109/CVPR.2016.90.
27. P. Agarwal, A. A. Rahman, P.-L. St-Charles, et al., “Transformers in reinforcement learning: A survey,” *arXiv*, 2023, doi: 10.48550/arXiv.2307.05979.
28. D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” *arXiv*, 2014, doi: 10.48550/arXiv.1409.0473.
29. Y. Yuan and L. Lin, “Self-supervised pretraining of transformers for satellite image time series classification,” *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 474–487, 2021, doi: 10.1109/JSTARS.2020.3036605.
30. X. Ni, J. Li, M. Yu, et al., “Generalizable resource allocation in stream processing via deep reinforcement learning,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 04, pp. 857–864, 2020, doi: 10.1609/aaai.v34i04.5470.
31. L. Chen, K. Lu, A. Rajeswaran, et al., “Decision Transformer: Reinforcement learning via sequence modeling,” *Adv. Neural Inf. Process. Syst.*, vol. 34, pp. 15084–15097, 2021.
32. L. Song, Y. Li, and J. Xu, “Dynamic job-shop scheduling based on transformer and deep reinforcement learning,” *Processes*, vol. 11, no. 12, pp. 3434, 2023, doi: 10.3390/pr11123434.
33. F. Li, S. Lang, Y. Tian, et al., “A transformer-based deep reinforcement learning approach for dynamic parallel machine scheduling problem with family setups,” *J. Intell. Manuf.*, 2024, doi: 10.1007/s10845-024-02428-3.
34. S. Halder, K. H. Lim, J. Chan, et al., “Capacity-aware fair POI recommendation combining transformer neural networks and resource allocation policy,” *Appl. Soft Comput.*, vol. 147, pp. 110785, 2023, doi: 10.1016/j.asoc.2023.110785.
35. A. Manchin, E. Abbasnejad, and A. van den Hengel, “Reinforcement learning with attention that works: A self-supervised approach,” in *Proc. Int. Conf. Neural Inf. Process. (ICONIP)*, pp. 223–230, 2019, doi: 10.1007/978-3-030-36808-1-25.
36. A. Jaegle, S. Borgeaud, J.-B. Alayrac, et al., “Perceiver IO: A general architecture for structured inputs and outputs,” *arXiv*, 2021, doi: 10.48550/arXiv.2107.14795.
37. J. Ceron and P. Castro, “Revisiting Rainbow: Promoting more insightful and inclusive deep reinforcement learning research,” in *Proc. Int. Conf. Mach. Learn. (ICML)*, pp. 1373–1383, 2021.
38. J. Kaplan, S. McCandlish, T. Henighan, et al., “Scaling laws for neural language models,” *arXiv*, 2020, doi: 10.48550/arXiv.2001.08361.
39. P. Henderson, R. Islam, P. Bachman, et al., “Deep reinforcement learning that matters,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 32, no. 1, 2018, doi: 10.1609/aaai.v32i1.11694.
40. Y. Tay, M. Dehghani, D. Bahri, and D. Metzler, “Efficient transformers: A survey,” *arXiv*, 2020, doi: 10.48550/arXiv.2009.06732.
41. T. Lesort, N. Díaz-Rodríguez, J.-F. Goudou, et al., “State representation learning for control: An overview,” *Neural Netw.*, vol. 108, pp. 379–392, 2018, doi: 10.1016/j.neunet.2018.07.006.
42. M. Laskin, A. Srinivas, and P. Abbeel, “CURL: Contrastive unsupervised representations for reinforcement learning,” in *Proc. Int. Conf. Mach. Learn. (ICML)*, pp. 5639–5650, 2020.
43. D. Ghosh and M. G. Bellemare, “Representations for stable off-policy reinforcement learning,” in *Proc. Int. Conf. Mach. Learn. (ICML)*, pp. 3556–3565, 2020.
44. T. Brown, B. Mann, N. Ryder, et al., “Language models are few-shot learners,” *Adv. Neural Inf. Process. Syst.*, vol. 33, pp. 1877–1901, 2020.
45. D. Hafner, T. Lillicrap, M. Norouzi, et al., “Mastering Atari with discrete world models,” *arXiv*, 2020, doi: 10.48550/arXiv.2010.02193.

46. K. Esslinger, R. Platt, and C. Amato, “Deep Transformer Q-Networks for partially observable reinforcement learning,” *arXiv*, 2022, doi: 10.48550/arXiv.2206.01078.
47. M. Shibu, P. Leffelaar, H. van Keulen, et al., “LINTUL3, a simulation model for nitrogen-limited situations: Application to rice,” *Eur. J. Agron.*, vol. 32, no. 4, pp. 255–271, 2010, doi: 10.1016/j.eja.2010.01.003.
48. A. de Wit, H. Boogaard, D. Fumagalli, et al., “Twenty-five years of the WOFOST cropping systems model,” *Agric. Syst.*, vol. 168, pp. 154–167, 2019, doi: 10.1016/j.agry.2018.06.018.
49. W. Wiertsema, “Obtaining winter wheat parameters for LINTUL from a field experiment comparing old and young winter wheat varieties and evaluating the progress by breeding,” Ph.D. dissertation, Plant Production Systems, Wageningen Univ., Wageningen, The Netherlands, 2015.
50. H. Berghuijs, V. Silva, A. Rijk, et al., “Catching-up with genetic progress: Simulation of potential production for modern wheat cultivars in the Netherlands,” *Field Crops Res.*, vol. 296, pp. 108891, 2023, doi: 10.1016/j.fcr.2023.108891.
51. V. Mnih, K. Kavukcuoglu, D. Silver, et al., “Playing Atari with deep reinforcement learning,” *arXiv*, 2013, doi: 10.48550/arXiv.1312.5602.
52. J. Schulman, F. Wolski, P. Dhariwal, et al., “Proximal policy optimization algorithms,” *arXiv*, 2017, doi: 10.48550/arXiv.1707.06347.
53. A. Raffin, A. Hill, A. Gleave, et al., “Stable-Baselines3: Reliable reinforcement learning implementations,” *J. Mach. Learn. Res.*, vol. 22, no. 268, pp. 1–8, 2021.