Truck Fleet Dispatching in Open Pit Mines Using Artificial Intelligence Methods

Roberto Noriega and Yashar Pourrahimian
Mining Optimization Laboratory (MOL)
University of Alberta, Edmonton, Alberta, Canada

ABSTRACT

Truck fleet dispatching plays a key role in reducing operational costs and fulfilling operational targets in open-pit mining operations and is subject to large uncertainty arising from the operating cycles of trucks and loading equipment. In this research, we propose a reinforcement learning-based truck dispatching system. Reinforcement learning is a machine learning area that deals with learning an optimal sequential decision-making strategy in an uncertain environment. An open-pit load-and-haul simulation is developed that also captures the truck movements and interactions in the shared road network, and a Neural Network is successfully trained to suggest truck dispatching decisions in real-time.

1. Introduction

A truck dispatching system aims to assign trucks to their next destination after completing an operating cycle. An open-pit is a complex and dynamic production environment where trucks haul material between mining faces and different material destinations, travelling across a shared haul road network. A truck dispatching system must provide real-time assignments as trucks complete their cycles, based on the current state of the mine, to contribute towards the shift production targets. The operating cycle of trucks, comprising of hauling empty towards a shovel, receiving its payload, hauling the payload, and dumping it at a given destination, is subject to operational uncertainties at every step. These arise from equipment characteristics, equipment and road condition, and operator skill, amongst other factors. Furthermore, traffic rules such as no overtaking and different equipment sizes and production rates make the truck dispatching problem challenging to model and solve in a real-time manner.

The problem is usually modelled and solved in two stages, as proposed by White and Olson (1986) [1] and Olson et al. (1993) [2], where a first or upper stage defines a simplified model of the open-pit production environment and uses linear programming (LP) to obtain shovel digging rates and tonnage of material to be moved between different paths between mining faces and destinations in the pit. A lower stage refers to the real-time dispatch of trucks in operations and is solved using a heuristic rule to match trucks to shovels to meet the path tonnages established in the upper stage section. New developments have focused on more complex optimization models and using simulations (Temeng et al. (1998) [3], Ta et al. (2005) [4], Topal and Ramazan (2012) [5], Ahangaran et al. (2015) [6], Moradi Afrapoli et al. (2018) [7], Moradi Afrapoli et al. (2019) [8], Moradi-Afrapoli and Askari-Nasab (2020) [9], Moradi-Afrapoli et al. (2021) [10], Mohtasham et al. (2022) [11], Pirmoradian et al. (2022) [12], Moradi-Afrapoli et al. (2022) [13], Mirzaei-Nasirabad et al. (2023) [14], Ghasempour and Moradi-Afrapoli (2023) [15]). For a more detailed review of truck dispatching systems, readers are directed to Moradi Afrapoli and Askari-Nasab (2019) [16]. However, approaches based on linear optimization have a limited ability to capture the dynamics of the pit

ISBN: 978-1-55195-503-2

operations requiring significant simplifications or a large model that is incurring in large computing times. Moreover, these models make decisions based on estimates of equipment activity times which are highly uncertain, and simulation models have only been used to evaluate the performance of the dispatching model rather than to integrate the uncertainty into the optimization process directly.

This paper proposes an artificial intelligence (AI) based truck dispatching agent trained in an open-pit simulation environment that captures all the equipment cycle and interaction uncertainties. This allows the agent to learn from the system's uncertainties and directly incorporate them into the optimization process. The truck dispatching agent is trained to meet target shovel digging target rates in tonnes-per-hour (tph) and target tonnage targets to be delivered at the different destinations during a production shift. Furthermore, the training process is linked with the monthly production schedule and projected pit development to ensure the truck dispatching model performs optimally for the next month of production.

The truck dispatching agent is trained using Reinforcement Learning (RL). RL is a branch of Machine Learning (ML) that implements a computational approach to learning an optimal policy, a decision-making strategy, through interactions with an environment (Sutton and Barto (2018) [17]). In an RL framework, an agent, an abstraction for the decision-maker, interacts with an environment at different time steps. At any time-step t where the agent must act, it observes the current state of the system, s_t, and makes an action, a_t based on it. The environment then responds to this action by transitioning into a new state in the next time step s_{t+1}, and providing a reward R_{t+1} for the agent (Figure 1).

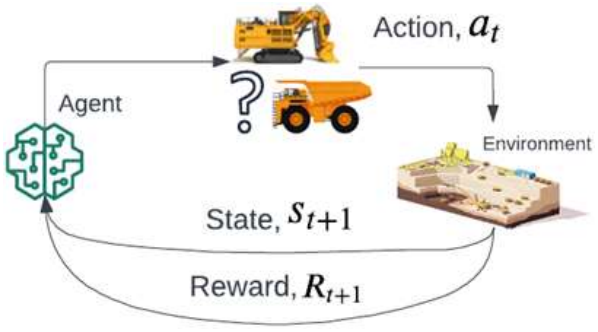

Figure 1. Reinforcement learning conceptual framework.

RL goal is to enable the agent to learn an optimal decision-making policy that maximizes the cumulative reward received throughout its interaction. The objective function that RL algorithms optimize is the total discounted reward accumulated by the agent by interacting with the environment. The RL agent seeks to learn a relation, or a function, from state-action pairs, (s, a), to the expected return of taking action a from state s, denominated Q-value, $Q(s, a)$. The Q-value is a function defined as the expected cumulative discounted reward achieved by the agent by taking an action a from a state s, following the same function for the next steps until the end of an episode (Equation 1). The cumulative nature of the return allows the agent to understand the impact of actions over long sequences rather than acting greedily at every step.

$$Q(s,a) = E\left[\sum_n \gamma^n R_n \middle| s_t = s, a_t = a\right] \quad (1)$$

The use of Deep Neural Networks (DNN) as function approximators in RL frameworks to learn the Q-value function provides the ability to learn complex and non-linear relations between environment states, agent actions, and returns. This framework has been used to develop complex scheduling and routing systems for complex real-world industrial applications in manufacturing plants (Lian et al.

(2022) [18]), chemical production plants (Hubbs et al. (2020) [19]), and vehicle routing platforms (Qin et al. (2020) [20]).

Machine Learning and Reinforcement Learning frameworks have also been explored for truck dispatching in open-pit mining (Noriega and Pourrahimian (2022) [21], Nobahar et al. (2022) [22]). Zhang et al. (2020) [23] proposed a deep Q-learning model that maximizes the total fleet productivity in open-pit operations, considering a truck fleet of heterogeneous capacities. However, the system does not consider potential production targets at different destinations, making it impractical for real-world operations where ore and waste targets can vary depending on the different processing methods applied in the mine. Moreover, the system is trained and evaluated for a single production shift. de Carvalho and Dimitrakopoulos (2021) [24], on the other hand, proposed a deep Q-learning that can dispatch trucks to meet different ore and waste production targets; however, the simulation that is used to train the agent is relatively simple and would not be able to fully capture the dynamics of a real-world operation which could lead to significant discrepancies in a potential real-world application. Furthermore, the authors train their system for only three days of production, and the evaluation indicates that the performance significantly degrades after five days suggesting that the model would need to be continuously retrained. This paper aims to address these limitations by training the agent in a detailed open-pit simulation environment and linking the training process with the production schedule to obtain optimal and consistent performance for a month of production.

2. METHODS

A discrete event simulation (DES) model is used to build a simulated open-pit production environment to train the truck dispatching agent. DES models have been widely used to evaluate mining fleet productivity and the impact of changes to the mining production environment (Upadhyay et al. (2021) [25], Moradi-Afrapoli et al. (2022) [26]) and provide an approximation of the impact of truck dispatching decisions in the open-pit system state. The DES is built using information about the mine layout to model the haul road network, mining faces and destination locations, and other relevant features. Equipment dispatch and mine planning databases are used to model the uncertainties in equipment activities as probability distributions. The DES then simulates the movement of trucks and their interaction with shovels and dumping destinations keeping track of different key production indicators (KPIs) and productivity rates. Whenever a truck dispatching decision is required, control in the DES is yielded to the AI truck dispatching agent, which observes the current state of the mine, described as a vector, and outputs a dispatching action. The agent is a neural network (NN), trained by interacting multiple times with the environment, receiving a feedback signal, the Q-value cumulative discounted returns. The feedback signal is designed to encourage the agent to meet desired goals, in this case, maximizing equipment utilization and meeting the different shift production targets. Figure 2 presents the methodology proposed in this paper. Due to the extensive AI packages available, the DES model is implemented in Python, an open-source programming language. The training loop is executed using PyTorch's deep learning framework.

2.1. Open-pit environment

The open-pit DES environment simulates a shift of production where trucks move between shovels and dumping destinations across a shared haul road network. Stochasticity is considered in each component of the truck operating cycle: hauling empty, getting loaded, hauling loaded and dumping. With queues forming based on the limited serving capacity of both shovels and dumping destinations. The DES is based on the models described by (Goris Cervantes et al. (2019) [27]). At the start of each shift simulated, shovels are assigned to a mining face based on the mine plan which also dictates the destination of the material being loaded there. Truck dispatching actions are required every time a truck dumps its payload, where the system's current state is given as input to the AI dispatched to

obtain a truck-shovel assignment for its next cycle. Figure **3** shows the general logic of the DES environment and its interaction with the AI dispatcher agent.

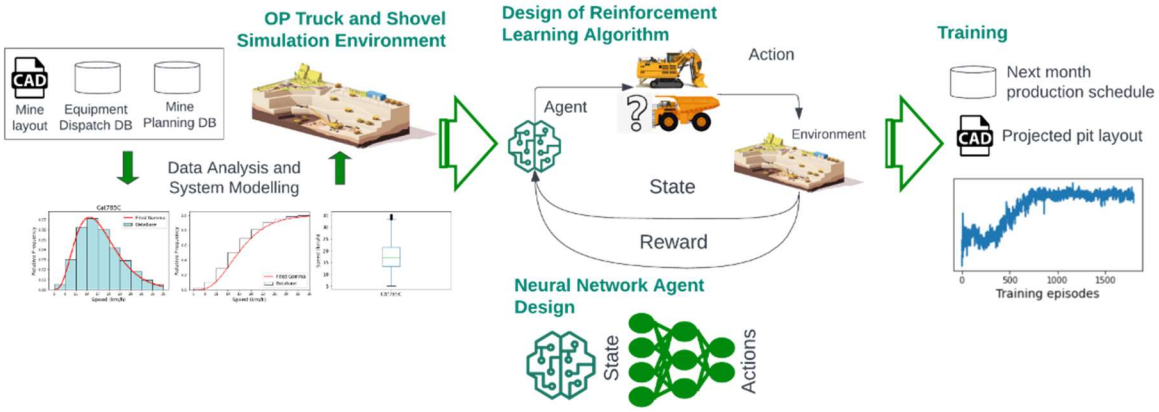

Figure 2. General methodology proposed for the development of an AI based truck dispatching agent.

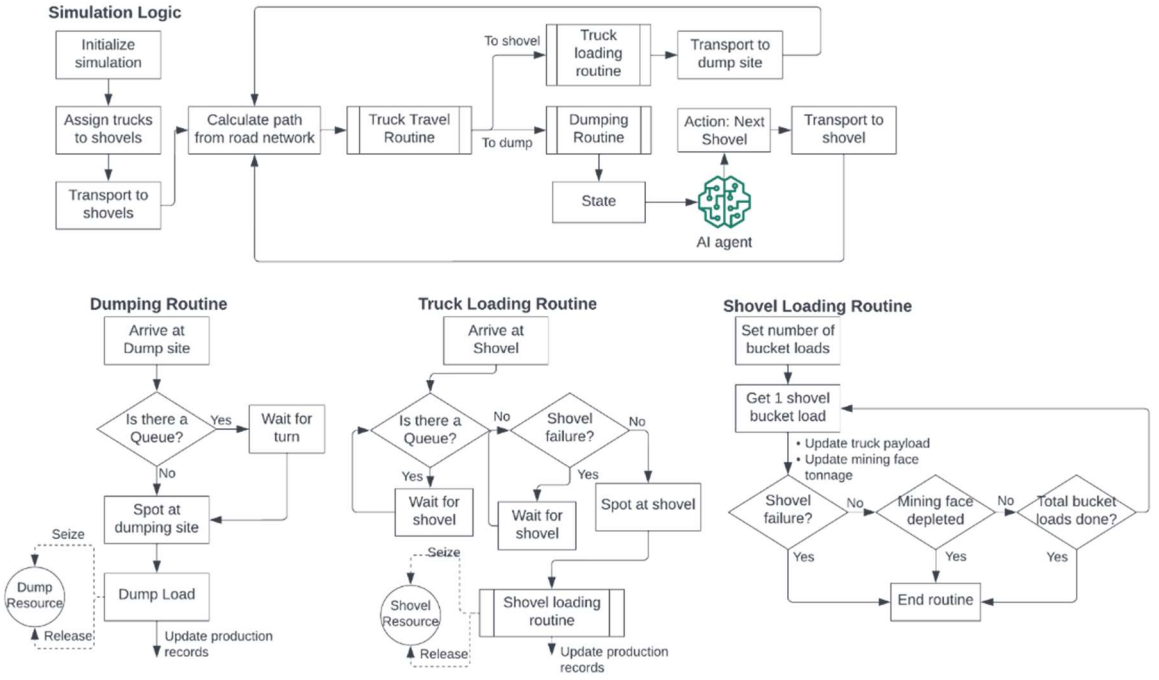

Figure 3. Interaction between the open-pit DES environment and the AI dispatcher agent.

When a truck starts traveling across the road network towards either a shovel or a dump destination, it is assigned a path, a collection of connected road segments from the network that is the shortest path between the current location of the truck and its next destination. Each road segment in this path contains information about the segment's rolling resistance, grade resistance, and length and maintains a queue of trucks that are traveling across it which is used to model truck bunching. Moreover, each segment has a maximum speed limit v_{road} that constrains the speed of any truck traveling through it.

Each truck traverses its path by moving along each road segment. Truck velocities are estimated using the equipment dispatch database to model each truck type's empty flat haul movement distribution. This refers to the velocity of trucks when hauling empty over flat segments and provides a good baseline then to include the effects of payload and road resistance. After entering a segment, the truck enters the road segment truck's queue and is assigned a velocity based on its truck type

empty flat haul velocity distribution adjusted by the road resistance and payload based on the truck manufacturer's rimpull curves. While travelling through the different road segments, trucks are not allowed to overtake and bunch behind, adopting its leading truck's velocity.

2.2. Truck Dispatching Agent

The implementation and training of the AI dispatcher agent require the definition of state, action, rewards, and the NN architecture. The decision on these components is based on the goals that the AI agent must reach. In this paper, we design the agent to dispatch trucks to meet given shovel digging rate targets for each shovel, expressed in tonnes-per-hour (tph), meet target tonnes dumped at the different destinations by the end of the shift and maximize equipment utilization.

To handle large truck fleets and improve generalization, each truck is considered an individual agent that collects state, action, next state, reward transitions, or experiences. However, a centralized truck dispatching policy is used to avoid incurring in training many individual agents, which severely suffers from scalability and implementation issues. Each truck pushes its experiences to a shared buffer that is used to train one general truck dispatching NN agent. Therefore, each transition between states of the pit encodes information about dispatching a single truck and its impact on the production environment after completing its cycle.

Considering an open-pit with m shovels and n destinations, the state of the system refers to an observation of useful features that the agent can use to correlate with the expected return of actions. In this case, the state is a vector that encodes information available by processing real-time data based on equipment tracking systems, production targets, and monitoring of shift progress. The AI agent's action is assigning the truck to a shovel for its next operating cycle. This action is encoded as a one-hot vector, a vector with a dimension equal to the number of shovels where the index or position of each element is tied to a shovel in the system. All its elements are 0 except for one element with a value of 1 where its position indicates the shovel assignment.

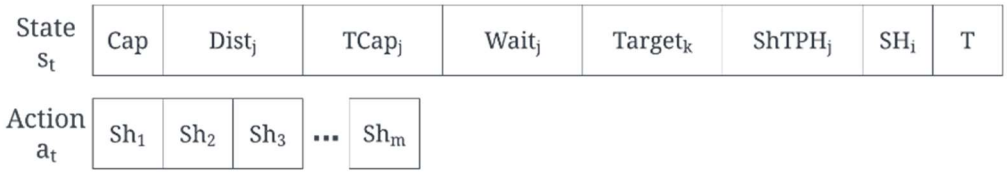

Figure 4. State and action representation for the truck dispatching agent.

In the state and action descriptuion the i, j, k indices refer to trucks, shovels and destinations respectively. The $Cap \in \mathbb{R}$ component is a real number indicating the capacity of the current truck being dispatched; this allows the agent to handle fleets with trucks of different sizes better. $Dist_j \in \mathbb{R}^m$ is a vector containing the total distance to be travelled along the road network for the next operating cycle between the location of the current truck and each shovel j. $TCap_j \in \mathbb{R}^m$ is a vector that contains information about the total truck capacity currently in route, at the queue and being serviced at each shovel, and its obtained based on the current truck's shovel assignments. $Wait_j \in \mathbb{R}^m$ elements contain the expected waiting for the current truck to be loaded at each shovel and is estimated based on path length, number of trucks in route, at the queue and being serviced by each shovel. $Target_k \in \mathbb{R}^n$ indicates the current progress of the target tonnage to be dumped at each destination. $ShTPH_j \in \mathbb{R}^m$ vector contains the current shovel's tph. $SH_i \in \mathbb{R}^m$ is a one-hot encoded vector that indicates the last shovel assignment of the current truck and $T \in \mathbb{R}$ is a real number indicating the current time of the shift. Each component is scaled to avoid the dominance of a single component due to large quantities.

The reward signal is designed to direct the agent to meet the desired goals. Once the agent dispatches a truck to a shovel and this completes its operating cycle by dumping its load at a destination point, a transition is completed and a reward, r_t, is provided as the truck waits for a new assignment. The reward function is shown in Equation 2.

$$r_t = \sum_j tph_j^- + (1 - qt_i) + \frac{Cap_i}{MaxCap}\delta_k + \delta_T \sum_k c_k^- \quad (2)$$

In the first component, tph_j^- indicates the current negative deviation, or shortfall, from the target tph of shovel j, expressed as a ratio. Adding all shovel production shortfalls encourages the agent to keep all shovels at the target rates, or higher. The second component rewards the agent for obtaining efficient truck cycles, where qt_i indicates the fraction of time the current truck spent at queues both at shovel and destination in its last cycle because of the action taken. The third component rewards the agent for every truck payload delivered, weighing by how large relative to the maximum truck size, where Cap_i is the capacity of the current truck, $MaxCap$ is the maximum truck capacity in the system and δ_k is a binary indicator for meeting the tonnage target to be delivered at the destination where the current truck dumped. This reward component only applies to destinations whose targets have not been met, using the binary indicator, to create a reward gradient with respect to other destinations with unmet targets and encourage the agent to meet all production targets. Finally, the last component of the reward is only applied at the end of an episode, where δ_k is a binary indicator that takes the value 1 when the shift ends, and penalizes the agent for not meeting the shift targets at each destination, where c_k^- is the shortfall from the target tonnes delivered at destination k at the end of the shift, expressed as a ratio.

The agent is implemented as a neural network (NN), which takes a state observation and estimates the Q-value for each action; that is, the expected discounted cumulative reward for taking each dispatching action from the given state up to the end of the shift. The action with the highest Q-value is then selected. The NN implemented in this paper is a feedforward network with five layers and 200 neurons in each layer using the ReLU function as the activation function.

2.3. Training Process

The AI dispatching agent is trained using the Deep Q-learning algorithm described by Mnih et al. (2015) [28], with the Double Q-learning modification proposed by van Hasselt et al. (2016) [29]. The training process refers to the iterative adjustment of the agent's NN weights, θ, to better predict the action-values, Q-value, for taking each action from a given state of the system.

The training process in deep Q-learning relies on the agent interacting with the environment storing experience vectors, $e_t = (s_t, a_t, r_t, s_{t+1})$ that represent each transition observed, in a memory replay buffer which serves as the training dataset for every training update of the NN agent. The role of the replay buffer is to break the correlation between consecutive experiences obtained from sequential time steps in the environment. The Q-values are estimated recurrently from the experience vectors by calling the NN to estimate the Q-value of the next state from the transition.

At every NN training step, a batch of experiences is drawn randomly from the replay buffer, and the NN weights are trained to minimize the prediction loss $L_i(\theta_i)$ defined as the mean square error (MSE) between the observed return (target) and the predicted return from the network at training step i, as shown in Equation 3.

$$L_i(\theta_i) = (r + \gamma max_{a_{t+1}} Q(s_{t+1}, a_{t+1}; {\theta_i}^{tgt}) - Q(s_t, a_t; \theta_i))^2 \quad (3)$$

Where θ_i^{tgt} refers to a NN used to evaluate target returns not trained at every step but synced with the online network defined by θ_i every C steps, called the target network, which helps stabilize the training process. The original Q-learning tends to overestimate the value of actions, which is harmful for training stability. Double Q-learning alleviates this problem by making the simple modification of using the training NN to choose actions but evaluating the next state Q-values using the target network and has been found to overperform the original implementation. Stochastic gradient descent (SGD) is used to update the NN weights to minimize the loss function. This process is controlled by a learning rate parameter, α, that scales the magnitude of the update in the direction of the loss gradients.

A training episode consists of a production shift and the agent's NN is updated at each transition, a truck operating cycle, as described above. To develop a dispatching model for the next month of production, at the start of the training shift the location of the shovels and the mine road layout is chosen randomly based on the production schedule and the projected pit development. That is the shift to be simulated and used for training is representative of one of the projected shifts for the next month. This helps avoid prohibitively long training episodes, as each episode is still one shift, and learn a general dispatching policy as the policy learned must be optimal for the whole month.

3. Results

3.1. Case Study

The AI truck dispatching agent was tested in a case study based on an iron ore mining operation. The mining operation uses a total of five shovels to load material from mining faces: 2 Hitachi 2500 shovels for ore production and 3 Hitachi 5500 Ex shovels for waste production. A fleet of 38 trucks is employed to haul the material from the pit to their destination, either one of two crushers or a waste dump. The mine uses 22 CAT785C, with a payload of 140 tonnes, and 16 CAT793C, with a payload of 218 tonnes. Two crushers are available, which both require 25,000 tonnes of ore to be delivered every shift. Figure 5 shows a plan view of the mine layout, in addition to the crusher and waste dump locations and the access to the mining faces areas. From the mining faces access, it is assumed that the distance to each mining face is the linear distance between its digging coordinate and the closest access point in the road network.

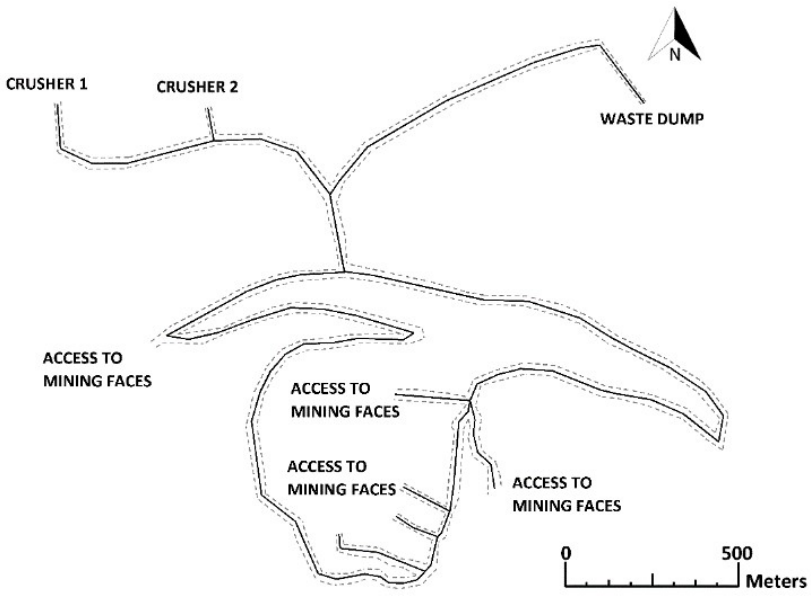

Figure 5. Case study iron ore open-pit layout.

Equipment activity records are available in the form of a Leica Jigsaw equipment dispatch database. This database is used to fit probability distributions to each shovel, truck operating activities, and truck haul speeds to build the stochastic DES open-pit for the case study. This probability fitting section is considered out of scope, not presenting any novelty, and not included to avoid a lengthy paper.

3.2. Training Results

The dispatcher agent is trained by interacting with the environment multiple times, simulating production shifts randomly based on the production schedule and pit layout. The agent's NN is updated to identify the best actions to take to maximize the total cumulative reward obtained at the end of the simulated shift. An epsilon greedy exploration strategy was used to encourage the agent

to explore and discover better dispatching strategies. When an action is required, there is a probability, ε, that the agent will select a completely random action. This probability starts high but decreases linearly as training progresses and the agent has experienced more episodes. The parameters used for the training of the AI truck dispatcher are shown in Table 1. The ε probability of taking random actions starts at 80% and decreases linearly to 1% during 500,000 environment transitions.

Table 1. Parameters used for the training of the truck dispatching agent.

Replay buffer size	200,000
Batch size for NN training updates	32
Discount factor, γ	0.99
Learning rate, α	5×10^{-6}
Iteration update frequency of target network	25,000

The system was trained in Google Colab platform, which provides powerful GPU cloud nodes to develop deep learning models. The training took about 4 hours and 20 minutes until convergence was achieved in the total cumulative reward obtained by the agent. Figure 6 shows the training performance, total production, average truck fleet cycle times, and average truck utilization as the agent learns an optimal dispatching policy.

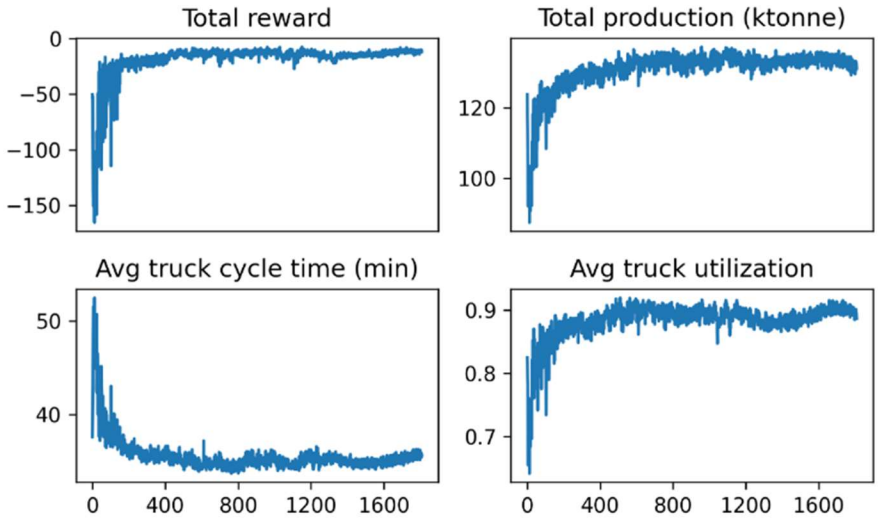

Figure 6. Training performance of the AI truck dispatching agent.

As training progresses, the agent improves its performance as measured by the total cumulative reward obtained at the end of each episode. After about 1000 training episodes, the agent can consistently achieve a high total cumulative reward over all simulated shifts randomly drawn considering the next production month. Since the agent receives a penalty for failing to keep the shovels at the target tph, total negative rewards indicate irregular shovel production. By the end of the training, the agent maintains a consistent production at all shovels, as its total cumulative reward approaches zero, where the small shortfall is caused due to the first hours of production when the trucks are starting their initial trips. The improvement in the performance of the agent can also be observed in the total production obtained, the improvement in the average truck cycle times and truck fleet utilization.

3.3. Performance Evaluation

During the training process, the performance of the dispatcher was measured based on a single shift drawn from the projected configurations of the pit during the next month. To evaluate the performance of the AI dispatcher, the entire month is simulated in one run. The DES is modified to allow shovel movement between mining faces as their current working areas are depleted. When this happens, a shovel is marked as unavailable, and the AI dispatcher is asked to re-dispatch all trucks currently assigned to it. Once the shovel arrives at its new working area, it is marked as available again.

The agent is tested for a month of production in the case study described above, using a mixed fleet of 22 CAT 785C and 16 CAT 793C trucks, 2 HIT 2500 shovels for ore production and 3 HIT 5500EX shovels for waste stripping. A total of 20 replications of the production month are used to obtain an average value for the different KPIs.

The agent managed to meet the shift production target and obtain extra daily production over the month as shown in Figure 7. The dips in material correspond to time slots where the shovels finish mining their current working area and move to the next one based on their schedule, where they are unavailable to constrain the system.

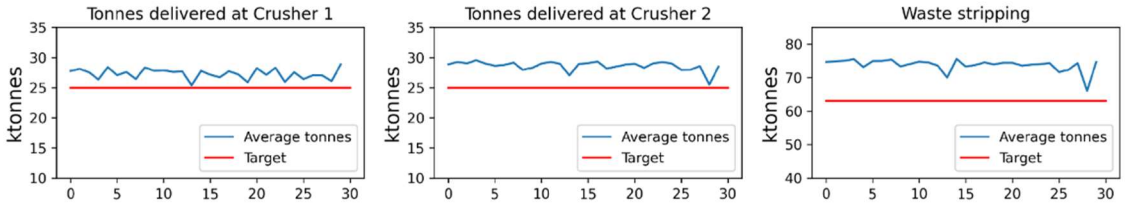

Figure 7. Average daily tonnes delivered at the destination over the month of production by the AI truck dispatching agent.

The AI agent also managed to meet the different shovel productivity rates, expressed as tonnage-per-hour (tph) loaded, as shown in Figure 8. The agent not only manages to maximize the total productivity of the system but also consistently meets the different shovel-loading targets at the different mining faces established from the mine plan, to ensure the projected advance in the ore mining faces and waste stripping requirements.

Moreover, the truck dispatching also achieves high average utilization rates for the equipment fleet. Figure 9 shows the average utilizations achieved for the shovel and truck fleet. The shovel fleet achieves an average of 90% utilization across the month which indicates the agent manages to keep all the shovels reasonably busy throughout the shift. The truck fleet utilization hovers around 85%, which could indicate some inevitable queues formed at the shovels.

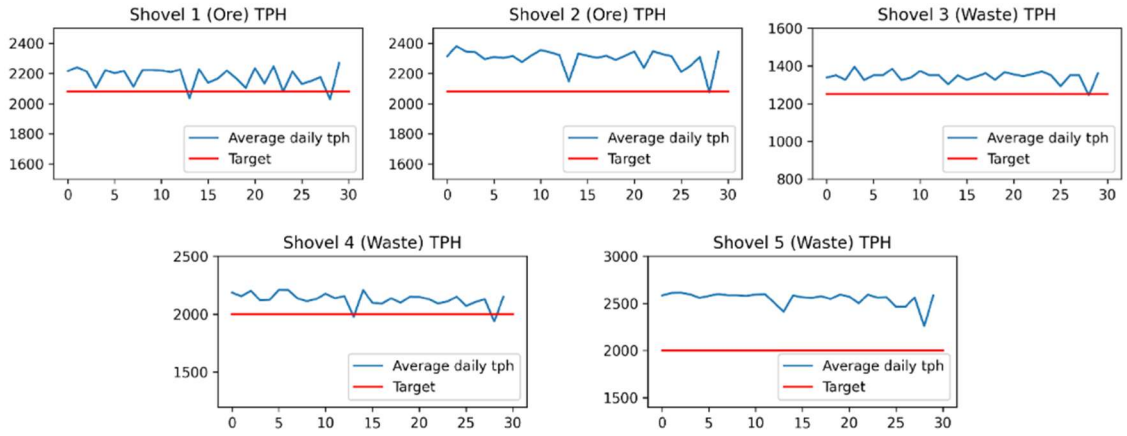

Figure 8. Average daily shovel tonnes-per-hour (tph) rate over the month of production obtained by the AI truck dispatching agent.

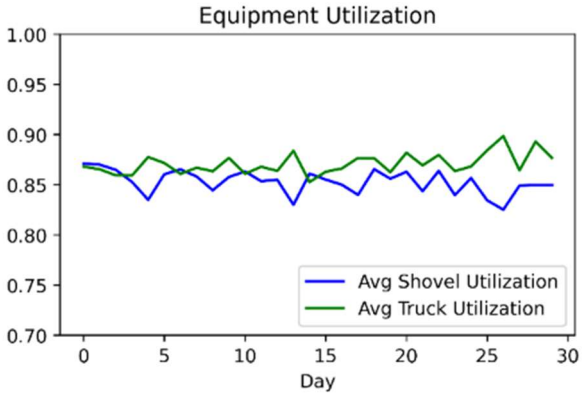

Figure 9. Average equipment fleet utilization achieved by the AI truck dispatching agent over the month of production.

4. Conclusions

This paper proposed an Artificial Intelligence (AI) based truck dispatching agent for open-pit mining. The agent is trained using a Deep Reinforcement (DR) Learning algorithmic framework, which has seen successful application in other production industries. The AI truck dispatcher is trained in a discrete simulation model of the open-pit operation, that considers stochasticity at every stage of the truck and shovel operating cycles. This allows the agent to learn the uncertainties in the behavior of the system and discover efficient dispatching strategies. The agent is trained to maximize equipment utilization and meet shovel productivity rates and destination rates, including ore delivered to crushers and waste stripping. After the agent is trained, obtaining a truck assignment is fast as it only requires a forward pass through a neural network, which takes a fraction of a second, making it appealing for real-time truck dispatching.

The framework is evaluated in an iron open-pit operation, where the simulation environment was built using the mine planning database, the mine layout, and the equipment dispatch database to model the stochasticity in the equipment activities. The agent was trained to learn a dispatching strategy for the next month of production and successfully met all the targets while maximizing equipment utilization and overall production.

The use of data-driven AI-based truck dispatching systems would allow the efficient use of multiple real-time data streams to take full advantage of new developments in monitoring systems for open-pit operations and digital twin technologies. Although the system currently takes about 4 hours and

a half to train for one month of production, future research will focus on extending the time over which the truck dispatching system operates and developing efficient training strategies to minimize computational effort—additionally, incorporating extra data streams such as equipment health and shovel bucket load sensors to provide better real-time dispatching solutions.

5. Rerences

[1] J. W. White and J. P. Olson, “Computer-based dispatching in mines with concurrent operating objectives,” Min. Eng., vol. 38, no. 11, pp. 1045–1054, 1986.

[2] J. P. Olson, S. I. Vohnout, and J. White, “On improving truck/shovel productivity in open-pit mines,” CIM Bull., vol. 86, no. 973, pp. 43–49, 1993.

[3] V. Temeng, F. Otuonye, and J. Frendewey, “A Nonpreemptive Goal Programming Approach to Truck Dispatching in Open-pit Mines,” Miner. Resour. Eng., vol. 07, no. 2, 1998.

[4] C. H. Ta, J. V. Kresta, J. F. Forbes, and H. J. Marquez, “A stochastic optimization approach to mine truck allocation,” Int. J. Surf. Mining, Reclam. Environ., vol. 19, no. 3, pp. 162–175, 2005, doi: 10.1080/13895260500128914.

[5] E. Topal and S. Ramazan, “Mining truck scheduling with stochastic maintenance cost,” J. Coal Sci. Eng., vol. 18, no. 3, pp. 313–319, 2012, doi: 10.1007/s12404-012-0316-4.

[6] D. K. Ahangaran, A. B. Yasrebi, A. Wetherelt, and P. Foster, “Real -time dispatching modelling for trucks with different capacities in open-pit mines,” Arch. Min. Sci., vol. 57, no. 1, pp. 39–52, 2012, doi: 10.2478/v10267-012-0003-8.

[7] R. Noriega and Y. Pourrahimian, “A systematic review of artificial intelligence and data-driven approaches in strategic open-pit mine planning”, Resources Policy, 2022, Vol. 77-102727

[8] A. Moradi Afrapoli, M. Tabesh, and H. Askari-Nasab, “A multiple objective transportation problem approach to dynamic truck dispatching in surface mines,” Eur. J. Oper. Res., vol. 276, no. 1, pp. 331–342, 2019, doi: 10.1016/j.ejor.2019.01.008.

[9] A. Moradi Afrapoli, S. Upadhyay, H. Askari-Nasab, "A Fuzzy Logic Approach towards Truck Dispatching Problem in Surface Mines ", International Journal of Mechanical and Production Engineering (IJMPE) , vol. 6, no.12, pp. 79-84, 2018.

[10] A. Moradi Afrapoli and H. Askari-Nasab, “A stochastic integrated simulation and mixed integer linear programming optimisation framework for truck dispatching problem in surface mines”, International Journal of Mining and Mineral Engineering, vol.11, no.4, 2020.

[11] M. Mohtasham, H. Mirzaei-Nasirabad, H. Askari-Nasab, and B. Alizadeh, “Multi-stage optimization framework for the real-time truck decision problem in open-pit mines: a case study on Sungun copper mine,” Int. J. Mining, Reclam. Environ., vol. 36, no. 7, pp. 461–491, 2022, doi: 10.1080/17480930.2022.2067709.

[12] A. Moradi-Afrapoli, S. Upadhyay, and H. Askari-Nasab. “Truck dispatching in surface mines -Application of fuzzy linear programming”, Journal of the Southern African Institute of Mining and Metallurgy, vol. 121, no. 9, pp. 505-512. https://dx.doi.org/10.17159/2411-9717/522/2021

[13] H. Pirmoradian, M. Monjezi, H. Askari-Nasab, E. Nikbakhsh, and A. Mousavi Nogholi. Development of a local search algorithm for solving allocation and dispatching problem of transportation fleet in open pit mines. Journal of Mineral Resources Engineering, 2022.

[14] A. Moradi Afrapoli, S. Prakash Upadhyay & H. Askari-Nasab, "A nested multiple-objective optimization algorithm for managing production fleets in surface mines", Engineering Optimization, 2022, DOI: 10.1080/0305215X.2022.2153840

[15] Mirzaei-Nasirabad, H., Mohtasham, M., Askari-Nasab, H. et al. An optimization model for the real-time truck dispatching problem in open-pit mining operations. Optimization and Engineering, 2023.

[16] A. Moradi Afrapoli and H. Askari-Nasab, "Mining fleet management systems: a review of models and algorithms," Int. J. Mining, Reclam. Environ., vol. 33, no. 1, pp. 42–60, 2019, doi: 10.1080/17480930.2017.1336607.

[17] R. Sutton and A. Barto, Reinforcement Learning An Introduction, 2nd Editio. MIT Press, 2018.

[18] Y. Liang et al., "Lenovo Schedules Laptop Manufacturing Using Deep Reinforcement Learning," Interfaces (Providence)., vol. 52, no. 1, pp. 56–68, 2022, doi: 10.1287/inte.2021.1109.

[19] C. D. Hubbs, C. Li, N. V Sahinidis, I. E. Grossmann, and J. M. Wassick, "A deep reinforcement learning approach for chemical production scheduling," Comput. Chem. Eng., vol. 141, p. 106982, 2020, doi: 10.1016/j.compchemeng.2020.106982.

[20] Z. Qin et al., "Ride-hailing order dispatching at DiDi via reinforcement learning," Interfaces (Providence)., vol. 50, no. 5, pp. 272–286, 2020, doi: 10.1287/INTE.2020.1047.

[21] M. Ghasempour Anaraki and A. Moradi Afrapoli, "Sustainable open pit fleet management system: integrating economic and environmental objectives into truck allocation", Mining Technology, 2023, DOI: 10.1080/25726668.2023.2233230

[22] P. Nobahar, Y. Pourrahimian and F. Mollaei Koshki, "Optimum Fleet Selection Using Machine Learning Algorithms—Case Study: Zenouz Kaolin Mine", Mining, vol. 2, no. 3, pp. 528-54, DOI: https://doi.org/10.3390/mining2030028

[23] C. Zhang, P. Odonkor, S. Zheng, H. Khorasgani, S. Serita, and C. Gupta, "Dynamic Dispatching for Large-Scale Heterogeneous Fleet via Multi-agent Deep Reinforcement Learning," arXiv:2008.10713v1, p. 9, 2020.

[24] J. P. de Carvalho and R. Dimitrakopoulos, "Integrating production planning with truck-dispatching decisions through reinforcement learning while managing uncertainty," Minerals, vol. 11, no. 6, 2021, doi: 10.3390/min11060587.

[25] S. Upadhyay, M. Tabesh, M. Badiozamani, A. Moradi Afrapoli, and H. Askari-Nasab, "A simulation-based algorithm for solving surface mines' equipment selection and sizing problem under uncertainty," CIM J., vol. 12, no. 1, pp. 36–46, 2021.

[26] A. Moradi-Afrapoli, H. Askari-Nasab, "Advanced Analytics for Surface Extraction". In: Soofastaei, A. (eds) Advanced Analytics in Mining Engineering. Springer, Cham., 2022, https://doi.org/10.1007/978-3-030-91589-6_8

[27] E. Goris Cervantes, S. Upadhyay, and H. Askari-Nasab, "Improvements to production planning in oil sands mining through analysis and simulation of truck cycle times," CIM J., vol. 10, no. 1, pp. 39–52, 2019.

[28] V. Mnih et al., "Human-level control through deep reinforcement learning," Nature, vol. 518, no. 7540, pp. 529–533, 2015, doi: 10.1038/nature14236.

[29] H. van Hasselt, A. Guez, and D. Silver, "Deep Reinforcement Learning with Double Q-Learning," in Proceedings of the AAAI Conference on Artificial Intelligence (AAAI-16), 2016, pp. 2094–20100, doi: https://doi.org/10.1609/aaai.v30i1.10295.