geosciences

MDPI

Article

Reinforcement Learning for the Face Support Pressure of Tunnel Boring Machines

Enrico Soranzo *, Carlotta Guardiani and Wei Wu

Institute of Geotechnical Engineering, University of Natural Resources and Life Sciences, 1180 Vienna, Austria
* Correspondence: enrico.soranzo@boku.ac.at

Abstract: In tunnel excavation with boring machines, the tunnel face is supported to avoid collapse and minimise settlement. This article proposes the use of reinforcement learning, specifically the deep Q-network algorithm, to predict the face support pressure. The algorithm uses a neural network to make decisions based on the expected rewards of each action. The approach is tested both analytically and numerically. By using the soil properties ahead of the tunnel face and the overburden depth as the input, the algorithm is capable of predicting the optimal tunnel face support pressure whilst minimising settlement, and adapting to changes in geological and geometrical conditions. The algorithm reaches maximum performance after 400 training episodes and can be used for random geological settings without retraining.

Keywords: tunnelling; tunnel boring machine; support pressure; face stability; reinforcement learning; machine learning; Deep-Q-Network

Citation: Soranzo, E.; Guardiani, C.; Wu, W. Reinforcement Learning for the Face Support Pressure of Tunnel Boring Machines. *Geosciences* **2023**, *13*, 82. https://doi.org/10.3390/geosciences13030082

Academic Editors: Franz Tschuchnigg, Georg H. Erharter, Thomas Marcher and Jesus Martinez-Frias

Received: 13 February 2023
Revised: 6 March 2023
Accepted: 8 March 2023
Published: 13 March 2023

1. Introduction

Face stability is critical in shallow tunnels to avoid collapse. In mechanised tunnels, the face support is provided by the tunnel boring machine (TBM), e.g., slurry (SPB), or earth pressure balance (EPB) shields. An estimate of the support pressure is required for safe and efficient construction.

The problem of face stability can be solved with analytical, numerical, and experimental approaches. The analytical methods are mainly based on the limit state analysis. Either the lower and upper bound theorems of plasticity in non-drained [1] and drained [2] conditions or the limit equilibrium method for SPB [3] and EPB [4] based on the original failure mechanism developed in [5] are used. These formulations have been recently considered in the design guidelines [6].

Face stability can also be investigated by numerical analysis. The finite element [7–9] and finite element limit analysis [10] methods have been used to obtain design equations. Alternatively, the problem can be studied experimentally at 1 g [8,11–15] or with centrifuge model tests [16–18]. Compared to 1 g tests, centrifuge model testing offers the advantages of stress and strain scaling due to the increased gravity field, thus increasing the accuracy in simulating complex soil-structure interactions.

Recently, machine learning has emerged as a promising technique for predictive assessment in geotechnical engineering, in general [19–21], and in tunnelling, in particular [22–25]. Some promising research domains for machine learning in tunnelling are the geological prognosis ahead of the face, the interpretation of monitoring results, automation, and maintenance [22]. At present, however, research appears to be focussed on the following topics: prediction of TBM operational parameters, penetration rate, pore-water pressure [26], ground settlement [27–29], disc cutter replacement [30–32], jamming risk [33,34], and geological classification [35–38]. Mainly due to the significant amount of data that is automatically collected by modern tunnel boring machines, the prediction of their operational parameters is a popular research topic. The researchers sought to reduce the misestimation

of the parameters [39], as well as to automatically determine the operational parameters based on geology [40] and to avoid trajectory deviations [41]. The prediction of the penetration rate is generally performed with respect to the geological and geotechnical site conditions. Ref. [42], by using, e.g., the cutter rotation speed, torque, and thrust force as input parameters [43], as well as the rock mass parameters and hydrogeological survey data [44,45]. Data filtering proved to improve the accuracy of the predictions [46]. Few authors estimated the face support pressure of TBMs with machine learning [24,47].

The previous studies developed models that are trained after tunnel construction and thus pertain to the domain of supervised learning, the machine learning paradigm where the predictions are based on labelled datasets [48]. This study proposes a method to determine the face support pressure of TBMs with reinforcement learning. In the field of artificial intelligence, reinforcement learning is a paradigm for addressing control problems where the actions taken by the algorithm are based on prior decisions. Reinforcement learning algorithms are taught by providing incentives to reach a goal. Although the study of reinforcement learning is still in its early stages, there have been some remarkable advancements, particularly those demonstrated by Google's DeepMind research team [49–53], including the ability to excel at playing Atari video games and board games, such as chess and go. The adoption of reinforcement learning from academia to industry is increasing. Notable examples include scheduling dynamic job shops [54], optimising memory control [55,56], personalising web services [57,58], autonomous vehicles [59], algorithmic trading [60], natural language processing [61] and healthcare applications such as dynamic treatment plans and automated medical diagnoses [62]. Industrial applications of reinforcement learning include Google Translate, Apple's Siri, and Bing's Voice Search [63].

Unfortunately, only a few of such studies are found in geotechnical engineering, especially in tunnelling. In particular, Erharter and Marcher presented a framework for the application of reinforcement learning to NATM tunnelling in rock [64] and to the TBM disc cutter replacement [30]. Zhang et al. [65] employed reinforcement learning to predict tunneling-induced ground response in real time.

In this study, the capability of the reinforcement learning algorithm to choose the best sequence of face support pressures is investigated by adapting the algorithm used by the DeepMind research group [51] and testing it on random geologies. The novelty of our method resides in the reinforcement learning approach, where the machine has no previous knowledge of the environment which it explores and is educated through the rewards defined by the user, as well as in the simulation of the environment with the finite difference method (FDM). This study shows that our model is capable of optimising the face support pressure, provided that a sufficient number of episodes are played.

The proposed method is outlined in the next section. The method is tested in environments of growing complexity, from analytical calculations to numerical analysis (Section 3). Its performance, limitations, and possible improvements are discussed in Section 4. Finally, Section 5 concludes the paper.

2. Methods

In this section, the reinforcement learning algorithm is described. It is implemented with the interpreted high-level general-purpose programming language Python [66]. The algorithm is tested against analytical calculations (Section 2.1) and numerical analysis (Section 3.4).

One of three basic machine learning paradigms alongside supervised and unsupervised learning, reinforcement learning involves learning optimal actions that maximise a numerical reward [67]. In other words, reinforcement learning deals with the way (the policy) an intelligent agent (an algorithm) takes the actions that maximise a user-defined reward in a particular setting (the state of the environment).

The policy defines how the algorithm behaves in a certain situation. More precisely, it connects the states of the environment to the actions to be taken. As such, the policy is the core of a reinforcement learning agent, given that it determines its behaviour [67]. The

well-established "epsilon greedy strategy", one of the oldest policies [67], is selected in this study. An action is considered "greedy" if it is expected to return the maximum reward for a given state. However, since the environment is unknown a priori, an initial exploration of the environment is necessary to determine these actions. This exploration begins with the first TBM excavation where the face support pressure is randomly chosen at every round. The randomness decreases after every episode as the environmental knowledge is exploited.

At each excavation round i, the agent chooses a random action n_a with probability ε and the action associated with the highest expected reward $Q_\mathrm{max}(S_{i+1}, a)$ with probability $1-\varepsilon$. ε is initialised at 1, which corresponds to a completely random selection, and is decremented by $1/N$ after each episode, where N is the total number of excavations (the episodes). For N episodes, ε decreases by $1/N$ per episode until it reaches 0. In mathematical terms, let $r \in \mathbb{R} \cap (0,1)$ be a random number and $\varepsilon = 1 - j/N$ at episode j, the agent takes the action A_i at the state S_i according to Equation (1), where $n_\mathrm{a} \in \mathbb{N} \cap (0,4)$ is a random integer.

$$A_\mathrm{i} := \begin{cases} Q_\mathrm{max}(S_i, a) & \text{if } r \geq \varepsilon \\ n_\mathrm{a} & \text{if } r < \varepsilon \end{cases} \tag{1}$$

Hence, the random choice of the face support pressure is initially the dominant pattern that is slowly, but steadily, abandoned over time as the agent gains some experience of the environment in which it operates. In other words, the face support pressure p_f becomes more of a "conscious" choice based on the rewards collected and represented by the value function $Q(s,a)$, which returns the reward expected for action a in the state s.

The rewards are user-defined and determined by the support pressure, the excavation rounds, and the surface settlement. They reflect the definition of efficient construction: a safe process (consisting of the minimisation of surface settlement) executed with the least possible effort (determined by the lowest possible support pressure). The actual reward values are irrelevant, as long as the algorithm can maximise the expected reward, given the input data. The relative weight of the rewards, however, has an impact on the results [68].

A $+1$ reward is collected at each excavation round. The lower the face support pressure applied by the TBM, the lower the building effort. Hence, a reward corresponding to $-\frac{p_\mathrm{f}}{200\ \mathrm{kPa}}$ is assigned to every action. The surface settlement at every round causes a -1 reward for each mm of additional settlement. A -100 reward is assigned if the surface settlement is larger than 10 cm (in the analytical environment) or if the calculations diverge (in the finite difference environment). These outcomes terminate the episode. The premature episode termination is called "game over" in reinforcement learning parlance. A $+100$ reward is assigned at excavation completion. Each completed or terminated tunnel excavation defines an episode. The rewards are listed in Table 1.

The "playground" of the agent is named environment [69]. The actions are chosen and the rewards are received by the agent in the environment. In the present case, two classes of environments are created. First, an environment consisting of simple analytical calculations of the required face support pressure and expected settlements is considered (Section 2.1). The second environment is simulated numerically via the FDM as described in Section 3.4.

Table 1. Summary of the rewards associated with the outcomes of the actions.

Group	Outcome	Reward	Episode Termination
Excavation	Round completed	$+1$	No
	Tunnel excavation completed	$+100$	No
Support pressure	Choice of the support pressure	$-\frac{p_\mathrm{f}}{200\ \mathrm{kPa}}$	No
Settlement	Additional settlement	$-1/\mathrm{mm}$	No
	Surface settlement >10 cm	-100	Yes
Numerical stability	Divergence of the calculation	-100	Yes

2.1. Analytical Training Environment

In the following, the initial training environment of the algorithm is described. For a tunnel with diameter $D = 10$ m, a random 2000 m long geological profile is generated (Figure 1). The pseudorandom number generator is set with a fixed seed to enable reproducibility of results. The 2000 m length corresponds to the break-even point for the choice of mechanised over NATM tunnelling [70]. At first, this geology is kept constant and is thus the same for all 100 training episodes. In the second instance, different random geological profiles are created at each episode (Section 3.2). The soil cover C is randomly initialised in the interval $(0.5, 3)D$. The soil's cover-to-diameter ratio is capped at $C/D = 3$. At every 2 m (the excavation step or round length), a random slope is selected from the interval $(-1, +1)$, corresponding to the interval of slope angles $(-45°, +45°)$. The soil unit weight γ, friction angle φ, cohesion c, and Young's modulus E are randomly initialised in the intervals shown in Table 2. The soil property values slightly change from their initial values at every excavation step. This means that, e.g., the unit weight at step $i + 1$ is calculated from the unit weight at step i as $\gamma_{i+1} = \gamma_i(1 + r_v)$ where r_v is a random number in the interval $(-1.25\%, +1.25\%)$. The soil properties are re-initialised with a 1% probability at every 2 m to simulate soil stratification.

As the agent navigates through the states of these environments, it gathers rewards and records them in the value function $Q(s, a)$. The state representation is described in the next section.

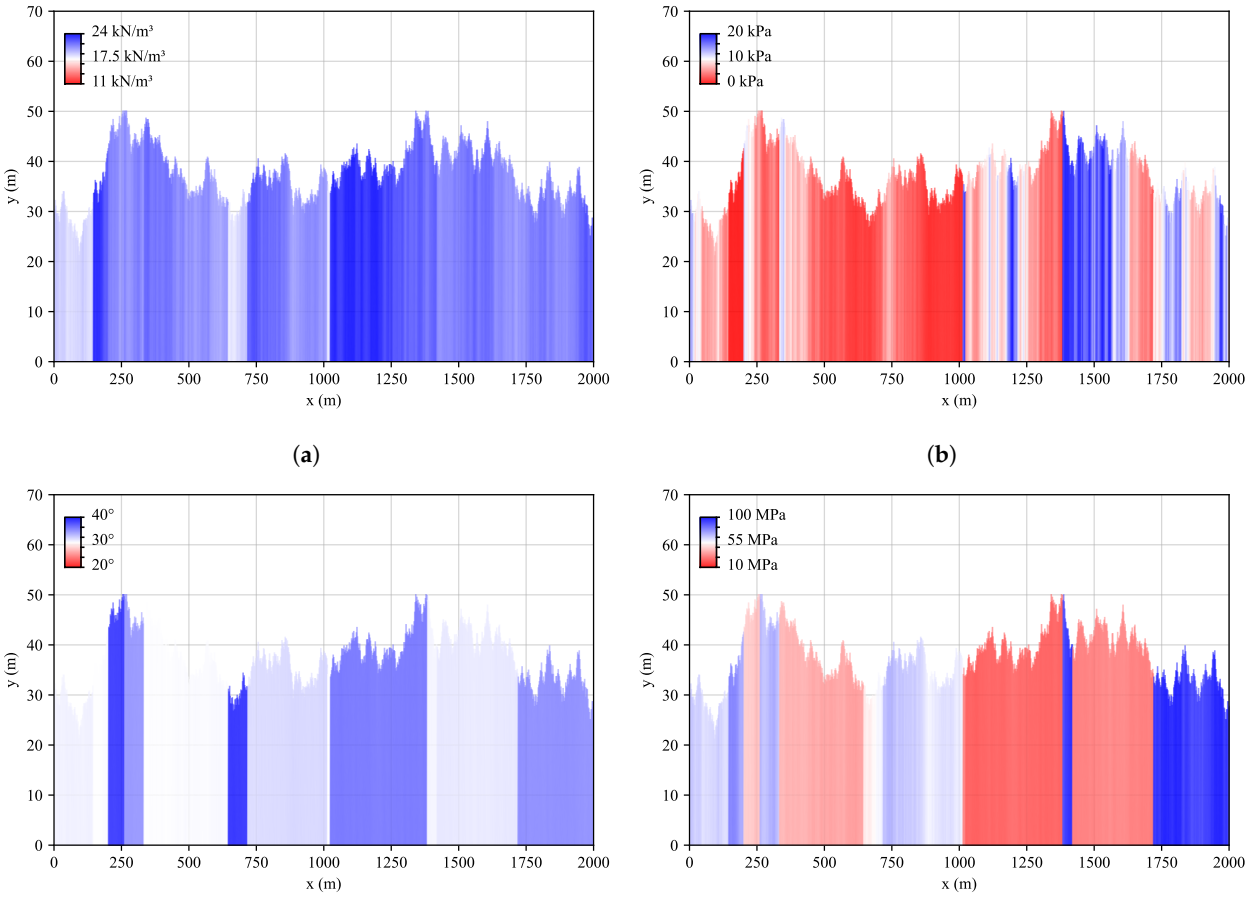


Figure 1. Random soil property values of the analytical environment with constant geology: (**a**) Unit soil weight. (**b**) Cohesion. (**c**) Friction angle. (**d**) Young's modulus.

Table 2. Range of the intervals of soil properties and their coefficient of variation for every 2 m excavation step.

Soil Parameter	Symbol	Unit	Minimum Value	Maximum Value	% Variation/m
Unit weight	γ	(kN/m³)	11	24	±1.25%
Cohesion	c	(kPa)	0	20	±10%
Friction angle	φ	(°)	20	40	±6%
Young's modulus	E	(MPa)	10	100	±1.25%

2.2. State Representation

The state represents the crucial and pertinent information needed to take an action. It is not the actual physical state of the environment, but rather a representation of the information used by the algorithm to make a decision [71].

Since there are no rules to determine the state variables, domain knowledge, i.e., "the knowledge about the environment in which the data is processed" [72], must be used. According to the literature [2,6,9,16,73], face stability primarily depends on the soil unit weight, cohesion, friction angle, and the depth of the overburden. Furthermore, soil settlement depends on the soil Young's modulus E and the stress release [74]. Hence, the soil properties γ, c, φ, E directly ahead of the tunnel face and the overburden C are normalised by dividing them by their maxima $\gamma_{\max}$, $c_{\max}$, $\varphi_{\max}$, $E_{\max}$, $C_{\max}$ (Table 2) and selected as the state variables. The stress release is determined by the choice of the support pressure.

Generally, TBMs cannot estimate the soil properties ahead of the tunnel face [75]. However, boreholes are retrieved prior to soil excavation and analysed in the laboratory to obtain the material properties for the engineering design. Unfortunately, due to the soil heterogeneity, the property values cannot perfectly match reality but are rather mean values that can be nonetheless used as a first approximation.

2.3. Face Support Pressure and Settlement

TBMs provide face support pressures up to approximately 200 to 300 kPa in soft soils [76]. At every excavation step, the proposed model searches the optimal support pressure within the interval $(50, 250)$ kPa. According to the guidelines [6], based on a limit equilibrium approach for drained conditions, the required support pressure is calculated with

$$p_{\mathrm{f,req}} = \frac{1}{A} \frac{(G + P_{\mathrm{V}})(\sin\vartheta - \cos\vartheta \tan\varphi') - 2T - c' \frac{D^2}{\sin\vartheta}}{\sin\vartheta \tan\varphi' + \cos\varphi'} \tag{2}$$

where $A = \pi \frac{D^2}{4}$ is the cross-sectional area of the tunnel, G is the self-weight of the sliding wedge, P_{V} is the vertical load from the soil prism, and T is the shear force on the vertical slip surface (Figure 2). The critical value of the sliding angle ϑ that maximises $p_{\mathrm{f,req}}$ is searched iteratively with the Python package `scipy` [77]. Note that the guidelines differentiate between c'_1 and φ'_1 above the tunnel and c'_2 and φ'_2 at the level of the tunnel. In this simulation, however, $c'_1 = c'_2 = c'$ and $\varphi'_1 = \varphi'_2 = \varphi'$.

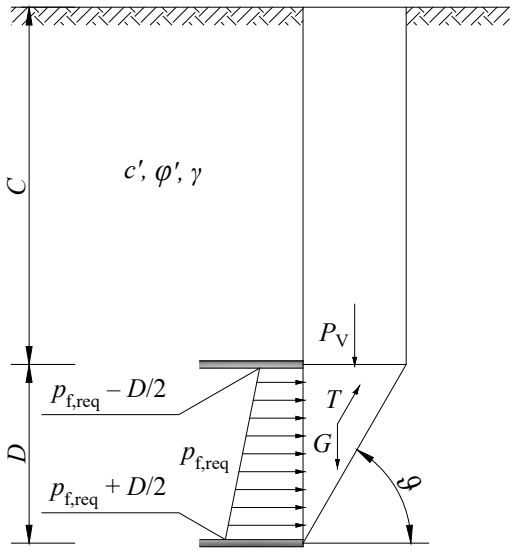


Figure 2. Tunnel face failure mechanism and forces acting on the sliding wedge according to [6].

The proposed model estimates the support pressure p_f at every step to maximise the expected outcome according to the following criteria and compares it with the required pressure $p_{f,req}$. In this simplified environment, the surface settlement occurs if $p_f < p_{f,req}$ and the stress release λ is calculated as the ratio of the provided to the required tunnel face support pressure, so that $\lambda = 1 - \frac{p_f}{p_{f,req}}$. The corresponding soil settlement u is then calculated as follows [74].

$$u = \frac{\lambda \gamma}{E} \left(\frac{D}{2} \right)^2 \tag{3}$$

In reality, the settlement occurs not only due to the stress release at the tunnel face but also due to other factors, such as the overcutting and ring gap [74]. Furthermore, an experience factor $K < 1$ depending on ground stress and conditions, and tunnel geometry is generally considered in Equation (3).

Since the model has no prior information about the required support pressure $p_{f,req}$, the support pressure is chosen randomly during the first episode. Then, the model learns the optimal support pressure based on the rewards collected, as explained in the next section.

2.4. Deep Q-Network

In this section, the algorithm to maximise the rewards collected during the TBM excavation is elucidated. Since the material properties have continuous values, the number of possible states as described in Section 2.2 is infinite. The model lacks complete knowledge of the expected rewards of actions at each state. Therefore, the knowledge of the value function is incomplete. To address this issue, the deep Q-network (DQN) is used.

DQN is a deep neural network that approximates the $Q(s, a)$ value function [78]. This algorithm, developed by Google research group DeepMind, was able to play six Atari games at a record level [49,51–53,79]. DQN is a specific approach to Q-learning, a method of learning optimal actions by predicting the expected reward associated with the state–action pairs, comparing the prediction to observed rewards, and updating the algorithm's parameters to improve future predictions. Formally, Q-learning algorithms are described by the following equation

$$Q(S_i, A_i) = Q(S_i, A_i) + \lambda_r [R_{i+1} + \Gamma \cdot Q_{max}(S_{i+1}, a) - Q(S_i, A_i)] \tag{4}$$

where $Q(S_i, A_i)$ is the reward associated with the state S_i and actions A_i, λ_r is the learning rate, R_{i+1} is the reward collected in the state S_{i+1}, Γ is the discount factor, and $Q_{max}(S_{i+1}, a)$ is the maximum reward in state S_{i+1}. The DQN is trained to choose the best tunnel face

support pressure based on state variables. It takes the state vector S_i as input and outputs the expected rewards for each action. After the agent takes an action A_i, the observed reward R_i is used to update the algorithm and improve future predictions. The algorithm is re-run using S_{i+1} as its input and returns the action with the highest value $Q_{\max}(S_{i+1}, a)$ of all the actions a. Afterwards, the learning algorithm is adjusted based on the actual reward. This is achieved by minimizing the mean squared error between the predicted and target prediction of $Q(S_i, A_i) + \lambda_r[R_{i+1} + \Gamma \cdot Q_{\max}(S_{i+1}, A) - Q(S_i, A_i)]$.

In Equation (4), λ_r and Γ are the hyperparameters that influence the algorithm learning process, namely the learning rate and the discount factor. Small updates are made by the algorithm at each step with a low value of λ_r and vice versa. The discount factor determines the extent to which the agent considers future rewards when making a decision. The higher is Γ, the more future rewards are taken into consideration.

The deep neural network architecture shown in Figure 3 is implemented with the `PyTorch` library [80] of the Facebook AI Research lab using the higher-level interface `nn`, based on [81]. The input layer is the state vector whose elements are $\gamma_i/\gamma_{\max}$, $c_i/c_{\max}$, $\varphi_i/\varphi_{\max}$, $E_i/E_{\max}$ and $C_i/C_{\max}$. The support pressure p_f is expressed in increments of 50 kPa in the range of natural integers $(50, 250)$ kPa. This results in an output layer with five elements, representing the five possible support pressures (50, 100, 150, 200 and 250 kPa) corresponding to the pressures typically provided by TBMs. The first and second hidden layers contain 50 neurons each. Although there are no strict rules on the optimal architecture of neural networks, a high number of neurons in each layer does not typically impair the performance, albeit requiring more computation. Moreover, the performance is generally better if the same number of neurons in all layers is used and if this number is higher than the size of the input layer. [82] The design of the neural network in this study was chosen based on these findings and on the previous applications to slope stability and tunnelling [25,83].

However, DQNs are known to suffer from training instability [49]. To address this, two common techniques, experience replay [84] and target memory [49], are used to stabilise the network as described in the following sections

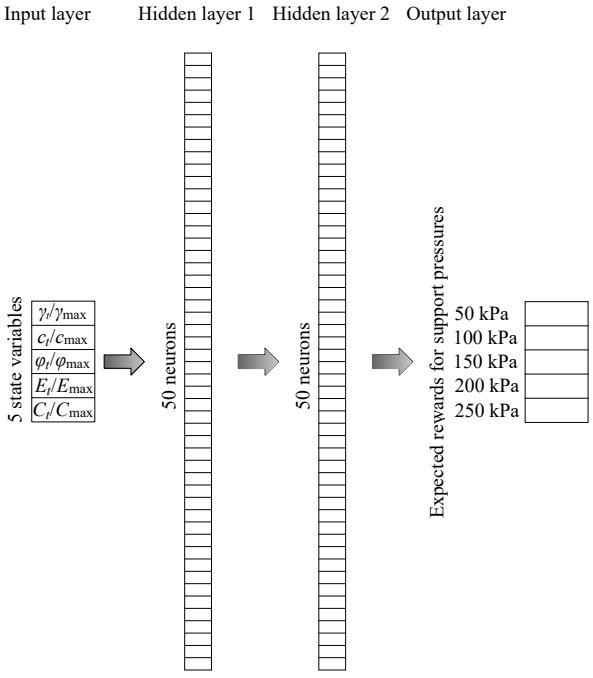


Figure 3. Architecture of the Deep Neural Network used for the choice of the support pressure based on the expected rewards and given the state of the TBM in the environment.

2.4.1. Experience Replay

Due to the varying soil properties, the algorithm cannot rely on memorizing a set sequence of support pressures p_f. It must determine the maximum support pressure that minimises the surface settlement, regardless of the geological conditions. However, reinforcement learning is a process that involves experimentation and can result in unstable solutions. This instability is particularly evident when rewards are "sparse", causing slow learning and leading to what is known as "catastrophic forgetting" [85]. This problem occurs in online training when backpropagation is performed after each state and is a common issue with gradient descent-based training methods. The root cause of catastrophic forgetting is the conflict between very similar state–action pairs, which hinders the ability to train the algorithm.

To improve the stability of the learning process in reinforcement learning, experience replay can be used. Experience replay accelerates the learning process by adding batch updating to online learning. The algorithm stores each experience, including the state, action taken, the resulting new state, and the reward in the list (s, a, s_{i+1}, r_{i+1}), until it reaches the specified "memory size" (s_m). Then, a random subset of the stored experiences, with a defined batch size (s_b), is selected for training. The algorithm calculates the value updates for each subset and stores them in target Y and state arrays X, which are then used as a mini-batch for training. Finally, the experience replay memory is overwritten with new values when it is full. Target memory is a further improvement of this technique.

2.4.2. Target Memory

The DQN instability is also caused by updating the parameters after each action, leading to a correlation between subsequent observations. To overcome this issue, the value function $Q(s, a)$ is updated only after a set number of episodes, rather than after each action, as suggested by [49].

In this scenario, the rewards are "sparse" and mostly higher at the end of each episode than after individual actions. To handle this, a duplicate of the Q-network, referred to as the target $\hat{Q}$-network, is created with its parameters lagging behind the original Q-network as described in [49]. The target memory is implemented by initialising the parameters θ_Q for the Q-network. The $\hat{Q}$-network, a copy of the Q-network, is created with distinct parameters θ_T that are, at first equal to θ_Q. Action a is selected with the Q-values of the Q-network by employing the epsilon-greedy strategy. The reward r_{i+1} and new state s_{i+1} are observed. The $\hat{Q}$-values of the $\hat{Q}$-network are set to r_{i+1} at the end of the episode or to $r_{i+1} + \lambda_r \cdot \hat{Q}_{max}(S_{i+1})$ otherwise. The $\hat{Q}$-value is not back-propagated through the $\hat{Q}$-network, but through the Q-network. The parameters θ_T are updated to θ_Q after a certain number of iterations, called synchronisation frequency (f). Up to this point, all the features of the algorithm have been presented. The complete workflow of the algorithm is depicted in Figure 4. In the next section, the results of the analytical environment are presented. The values of the hyperparameters λ_r, Γ, f, s_m, and s_b are chosen based on the sensitivity analysis of Section 3.1.

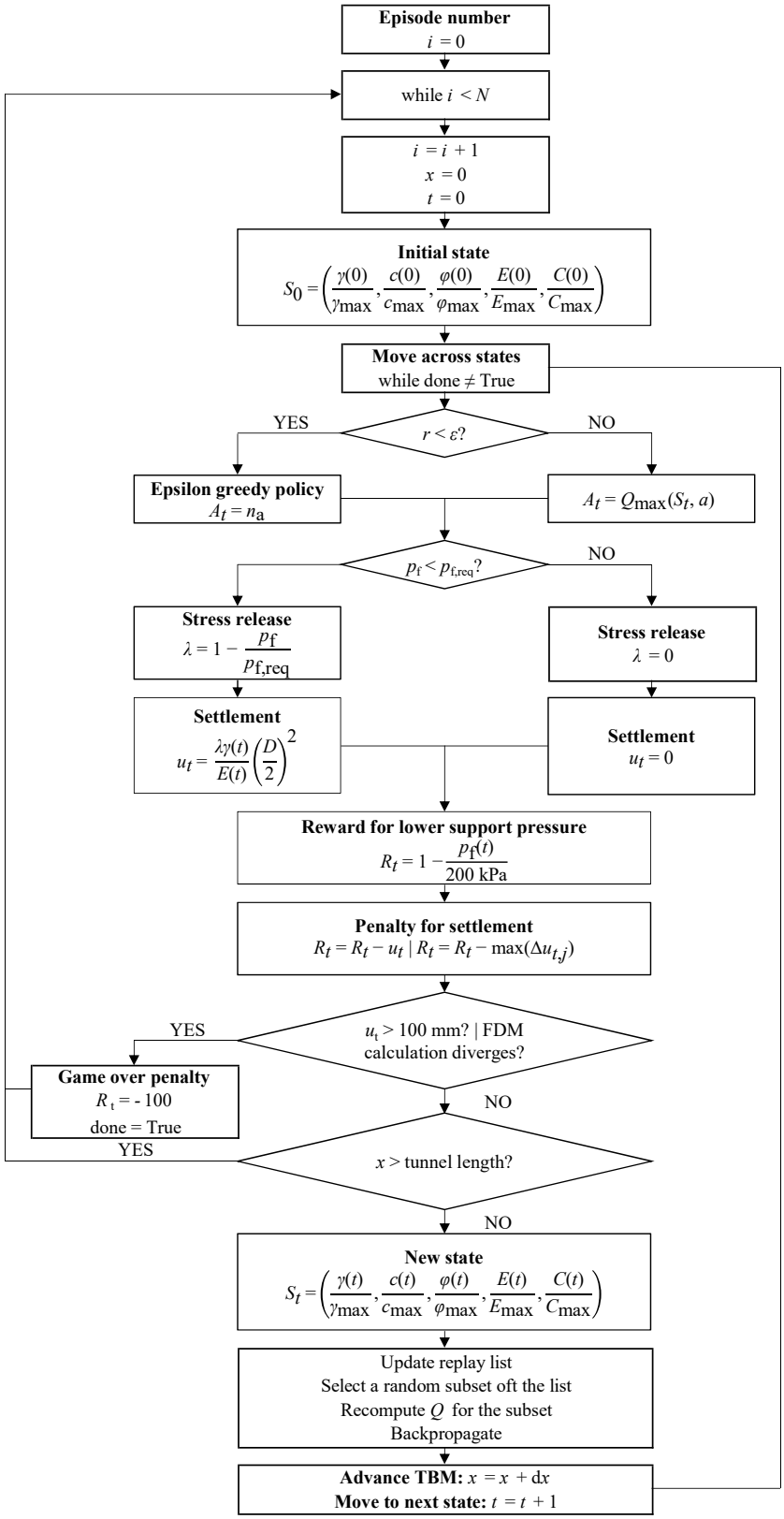
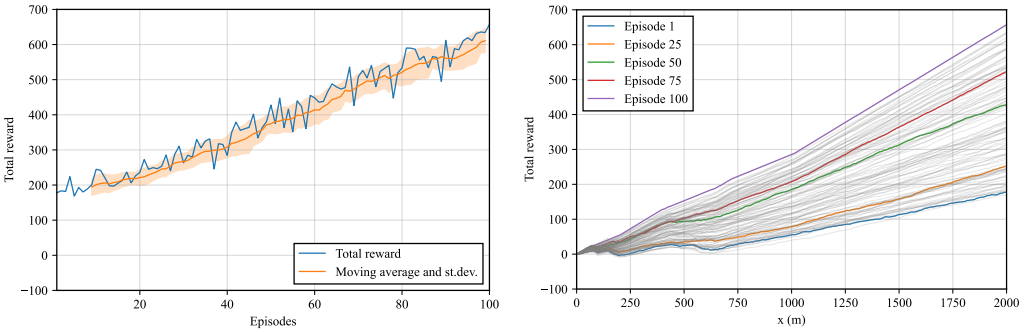


Figure 4. Workflow of the reinforcement learning algorithm.

3. Results of the Analytical Environment

In this section, the results of the analytical environment are presented. The cumulative reward collected at every episode is shown in Figure 5a. The orange line represents the

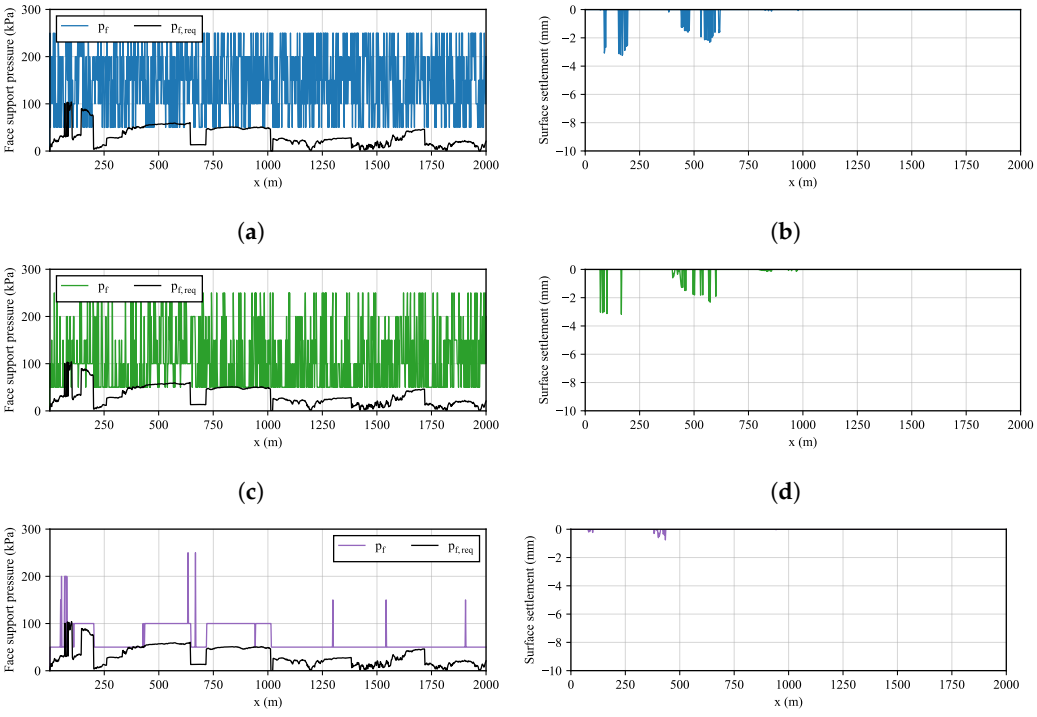
moving average of 10 episodes and the orange band is the range of the ±1 standard deviations. Since the algorithm is trained on the geology of Figure 1 only, the reward increases steadily, barely oscillating around the moving average. In Figure 5b, the cumulative rewards obtained from the 1th, 25th, 50th and 100th episodes are highlighted. As expected, the reward increases more rapidly for the last episodes. Moreover, approximately at the "chainages" $x = 200$ and 700 m, slight drops in the cumulative reward occur due to the abrupt changes in geology (Figure 1).



(**a**) (**b**)

Figure 5. Rewards of the analytical environment with constant geology: (**a**) Cumulative reward vs. no. of episodes; moving average and interval of range ±one standard deviation of ten episodes (orange band). (**b**) Cumulative reward vs. excavation step for all episodes.

The support pressures and the resulting settlement in episodes 1, 50, and 100 are shown on the left and right sides of Figure 6, respectively. As the support pressure is randomly chosen at the first episode, Figure 6a exhibits chaotic behaviour.



(**a**) (**b**)

(**c**) (**d**)

(**e**) (**f**)

Figure 6. Required and provided support pressure and settlement in the analytical environment with constant geology at episodes (**a**,**b**) 1, (**c**,**d**), 50, and (**e**,**f**) 100.

As the agent learns the optimal support pressures, this behaviour is mitigated and the support pressure becomes more stable (Figure 6c). Figure 6c also proves that the algorithm is able to choose the support pressure in such a way that it almost encases the required pressure. Furthermore, even when $p_f < p_{f,req}$, the resulting settlement is negligible (Figure 6f). The model sensitivity to changes in the values of the hyperparameters is studied in the next section.

3.1. Sensitivity Analysis

Contrary to the parameters of the deep neural networks depicted in Figure 3 that are learned by the algorithm, hyperparameters are set by the user. In Section 2, a number of hyperparameters have been introduced, such as the discount factor Γ, learning rate λ_r, synchronisation frequency f, the memory s_m, and batch s_b sizes.

The model sensitivity to these hyperparameters is summarised in Table 3. The analysis is performed starting from $\Gamma = 0.15$, $\lambda_r = 10^{-3}$, $f = 5$, $s_m = 10$, $s_b = 2$, and changing one hyperparameter at a time to lower and higher values. In doing so, the original combination of hyperparameters is shown to be the optimal one. Obviously, the results are not very sensitive to the discount factor and $\Gamma = 0.15$ returns the highest cumulative reward. A low value of Γ implies that the action taken at one state has little impact on the actions taken in the following states. This seems plausible for tunnelling where the settlement caused by an excessively low support pressure cannot be possibly offset by a later support pressure increase. The optimal learning rate is a matter of numerical stability. In this study, $\lambda_r = 0.001$ appears to be the optimal value and, if the learning rate is set at $\lambda_r = 0.01$, the cumulative reward cannot be properly maximised. The cumulative reward is not very sensitive to the synchronisation frequency where the optimal value is 5 episodes. The memory size has a more pronounced effect and the optimal value is 10. Finally, the batch size of 2 episodes provides the best results.

In summary, the proposed model is not very sensitive to the discount factor and the synchronisation frequency, it is starkly affected by the learning rate and the memory size in this particular problem. After the hyperparameters are optimised, the algorithm is tested against random geologies in the next section.

Table 3. Results of the analysis of sensitivity to the hyperparameters.

Hyperparameter	Values	Max. Reward
Discount factor Γ	0.01	621.0
	0.15	657.2
	0.2	622.2
Learning rate λ_r	10^{-4}	637.1
	10^{-3}	657.2
	10^{-2}	536.3
Synchronisation frequency f	5	657.2
	10	644.2
	15	652.1
Memory size s_m	5	647.3
	10	657.2
	15	562.8
Batch size s_b	5	630.0
	2	657.2
	1	609.3

3.2. Random Geologies

Prompted by the initial success in optimising the cumulative reward with constant geology, the algorithm is further tested against random geologies. Hence, the depth of the overburden and the material properties are no longer fixed as depicted in Figure 1, but

change at every episode. Figure 7 exemplarily shows the profiles of the cohesion values at episodes 2, 3, 4, and 5.

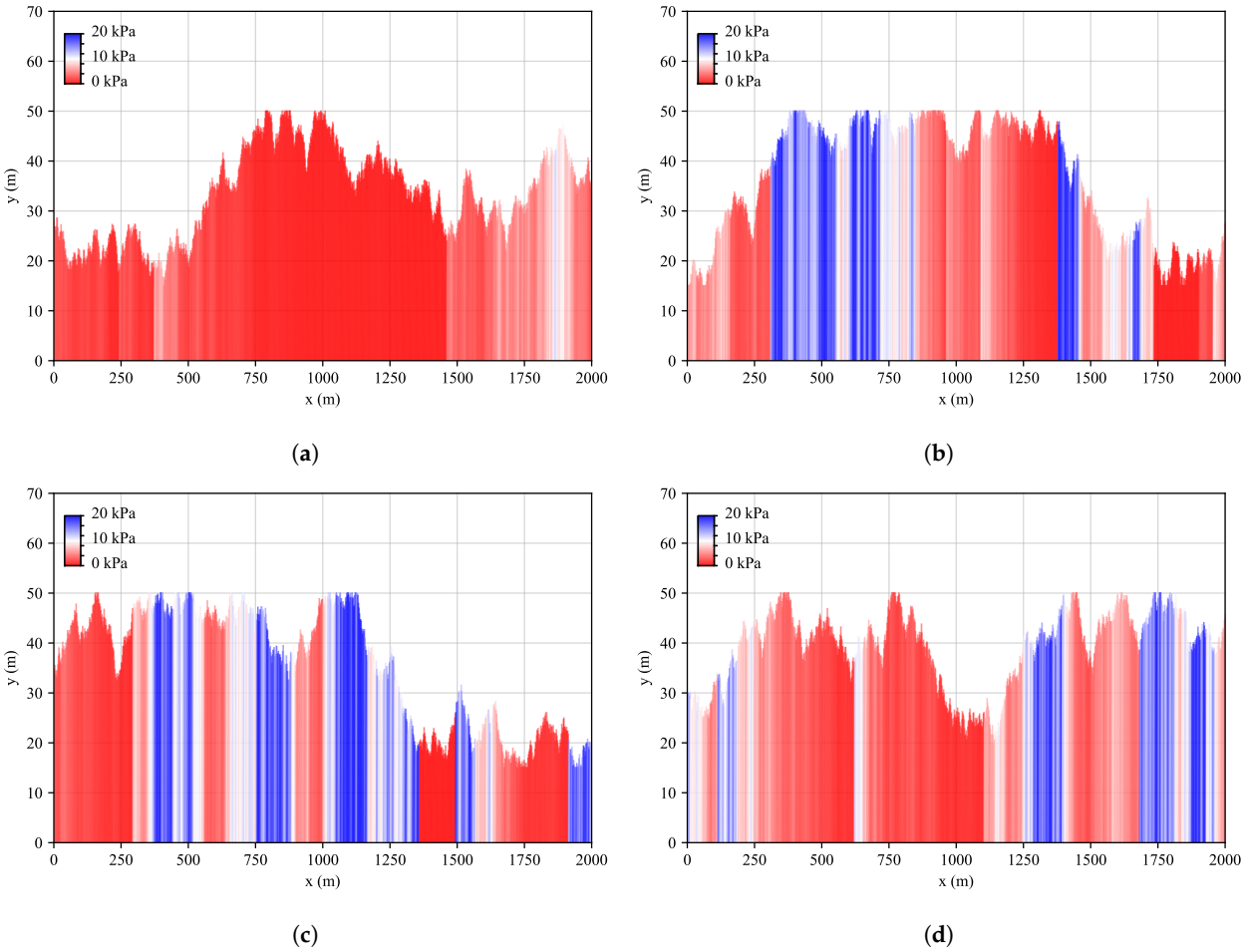
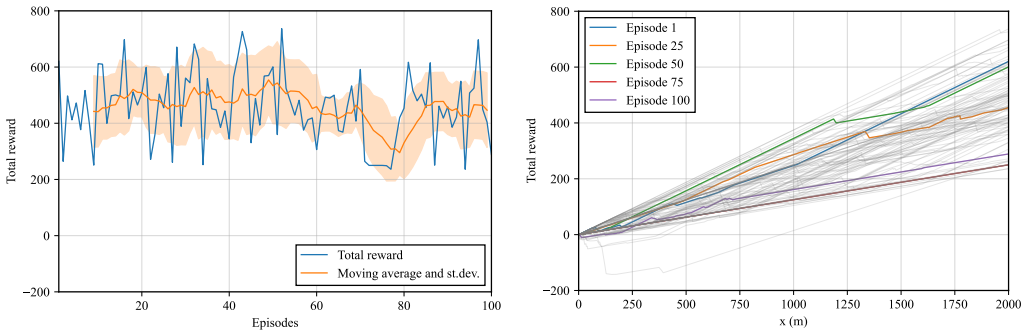


Figure 7. Cohesion values of the analytical environment with random geologies. (**a**) Episode 2. (**b**) Episode 3. (**c**) Episode 4. (**d**) Episode 5.

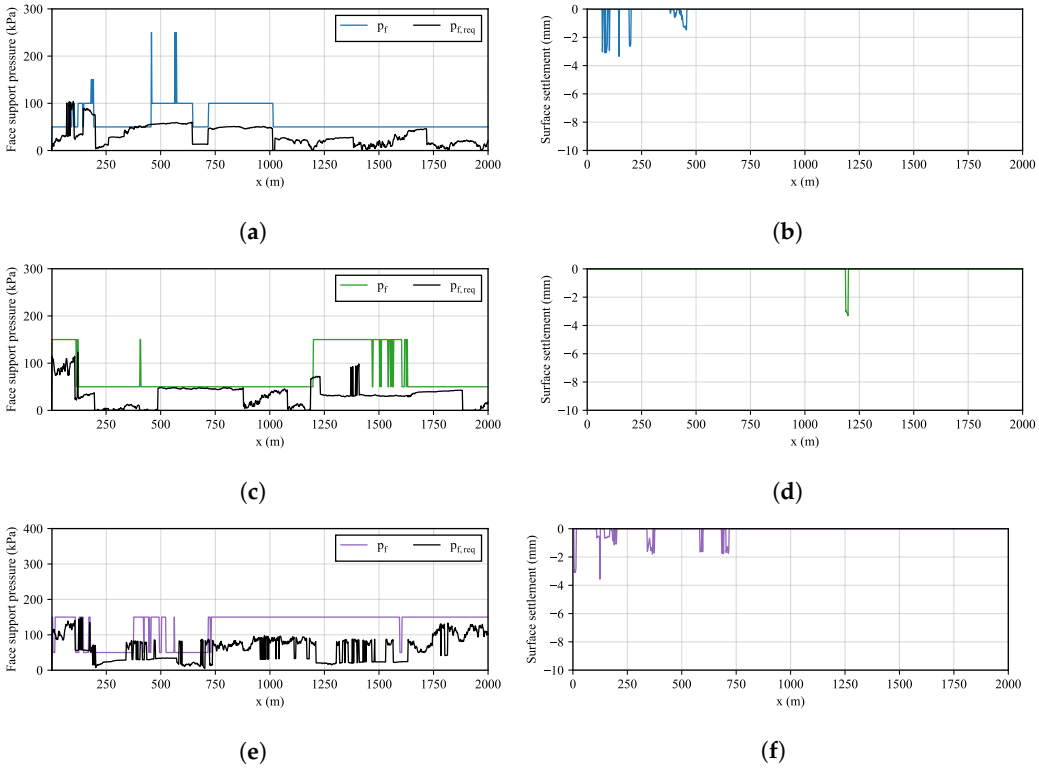
Clearly, since the environment changes at every episode, the cumulative reward shown in Figure 8a does not increase steadily as in Figure 5a, but swings markedly. The model is able to forecast the support pressure increases to accommodate for larger $p_{f,req}$ in different settings (Figure 9). Furthermore, even when the support pressure provided fall below the required one, the resulting settlement is lower than about 4 mm.

The previous results are obtained with no additional model training. This means that the initial value of ε is set to zero. By starting from different values of ε, it is shown in the following that no additional exploration is needed. Note that, strictly speaking, even if $\varepsilon_0 = 0$ the Q-value is slightly updated as rewards are collected along the way. The agent, however, would perform the same action, given a certain state, until its expected rewards fall below those of other actions. The results of this analysis are shown in Table 4 where the mean cumulative reward and standard deviation are listed for different ε_0. The highest mean cumulative reward and the lowest standard deviation are achieved for $\varepsilon_0 = 0$. Therefore $\varepsilon_0 = 0$ is selected as the optimum value, meaning that no additional exploration is required to optimise the algorithm performance in the random analytical environment. The numerical environment is described in the next section.



(**a**) (**b**)

Figure 8. Rewards of the random environment: (**a**) Cumulative reward vs. no. of episodes; moving average and interval of range ±one standard deviation of ten episodes (orange band). (**b**) Cumulative reward vs. excavation step for all episodes.



(**a**) (**b**)

(**c**) (**d**)

(**e**) (**f**)

Figure 9. Required and provided support pressures and settlement in the analytical environment with random geologies at episodes (**a**,**b**) 1, (**c**,**d**), 50 and (**e**,**f**) 100.

Table 4. Mean rewards and standard deviation of the random environment for different values of ε_0.

ε_0	Mean Reward	Standard Deviation
0.00	458.1	124.9
0.25	453.6	130.1
0.50	315.8	138.4
0.75	326.2	169.5
1.00	221.2	212.0

3.3. Effect of the Number of Episodes

When more episodes are played, the environment is explored more thoroughly. Hence, it is expected that the maximum cumulative reward increases with the number of episodes. Figure 10 shows that the maximum cumulative reward grows asymptotically up to approximately 683 after 400 episodes. It also shows that the maximum cumulative reward obtained after 50 episodes already represents approximately 90% of the maximum reward achieved after 400 episodes.

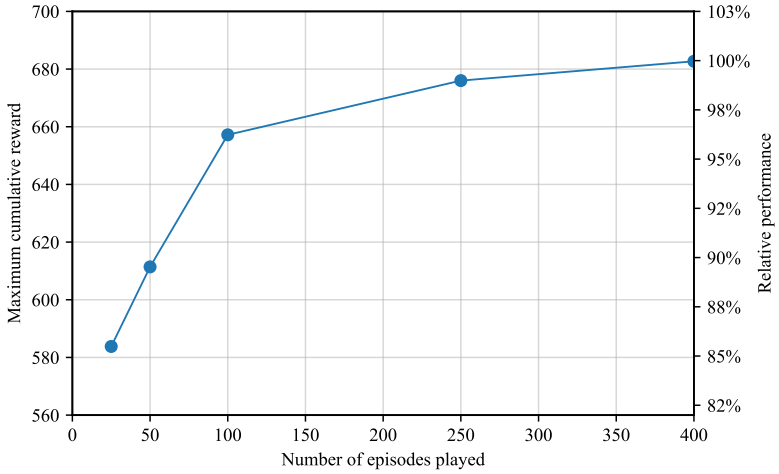


Figure 10. Effect of the number of episodes on the maximum cumulative reward.

3.4. Finite Difference Environment

Due to the simplified analytical formulations, the previous environment is not very realistic. It is, however, useful to demonstrate the capability of the model and to optimise its hyperparameters. Tunnels are often simulated with numerical analysis [86–89]. Hence, a more realistic finite difference environment is outlined in the following based on the finite-difference program FLAC^{3D} [90].

The mesh grid is 100 m wide, between 27 and 44 m high, and 100 m long. The tunnel diameter is 10 m and the distance between the tunnel axis and the bottom is 15 m. The soil surface slope changes at every 10 m of projected distance. The grid consists of 16,932 grid points and 15,300 elements with target dimensions of $2 \times 2 \times 1.5$ m (Figure 11). The linear elastic material law with Mohr–Coulomb failure criterion is considered. The soil properties are assigned within the intervals of Table 2 to randomly inclined layers. The property values are listed in Table 5. The displacements are fixed in the horizontal direction at the vertical boundaries and in the vertical direction at the bottom. The excavation is performed by removing the elements corresponding to the excavated soil. The support provided by the tunnel lining is simulated by fixing the displacement at the excavation boundaries. The support pressure is provided by applying a linearly increasing external pressure onto the tunnel face (Figure 12). This pressure is equal to 50, 100, 150, 200, or 250 kPa at the tunnel axis and increases linearly with depth according to the unit weight of the support medium $\gamma_{sm} = 12$ kN/m^3. The support medium consists of excavated soil and additives for EPB machines or slurry suspension for SPB and mix-shield machines [91]. The surface settlement is measured at the grid points situated on the surface every 10 m.

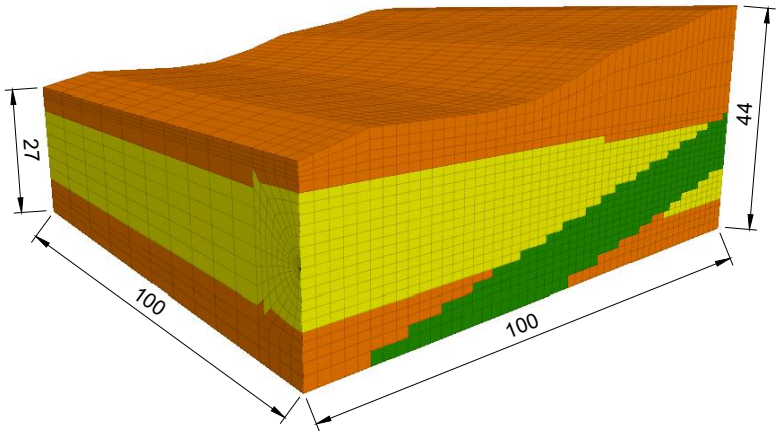


Figure 11. FLAC^{3D} three-dimensional model. The colours represent the randomly generated soil layers 1 (yellow), 2 (green), and 3 (orange). Units in metres.

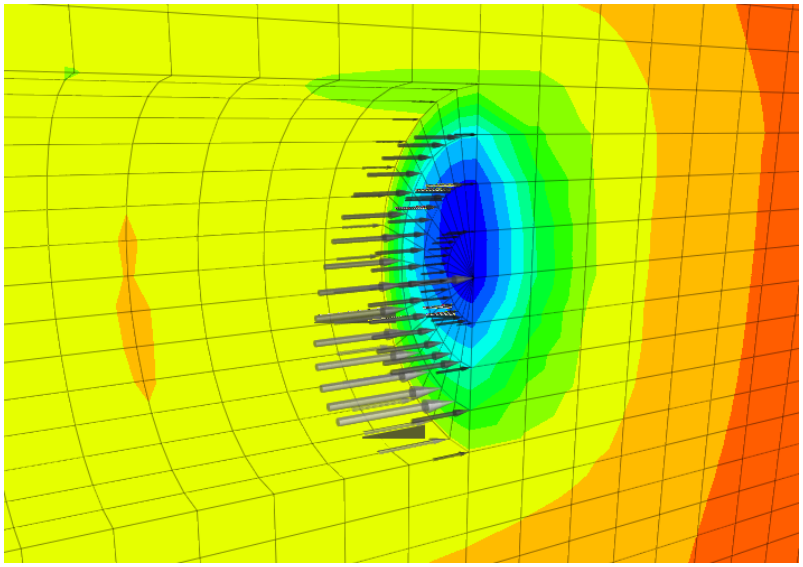


Figure 12. Detail of the linearly increasing support pressure at the tunnel face and the resulting horizontal displacement at a randomly selected excavation step.

Table 5. Soil property values assigned to the soil layers 1 (yellow), 2 (green), and 3 (orange) of Figure 11.

Soil parameter	Symbol	Unit	Layer 1	Layer 2	Layer 3
Unit weight	γ	(kN/m³)	23.0	13.7	15.9
Cohesion	c	(kPa)	14	1	11
Friction angle	φ	(°)	25	23	34
Young's modulus	E	(MPa)	11	32	13

In the analytical environment, the surface settlement is calculated at every excavation round. In reality, the cumulative surface settlement depends on the previous excavation steps. Hence, the settlement increase is considered in the finite difference environment to account for the effect of the tunnel face support pressure at each step. The settlement reward for state t is, thus, expressed as the maximum settlement difference between the excavation steps $\max(\Delta u_{t,j})$ of all of the j-th measuring points along the soil surface.

Finally, the game-over condition is not triggered by excessively large settlements, such as in the analytical environment, but by the divergence of the numerical solution.

Within the framework of transfer learning, the application of a machine learning model to a similar but unrelated problem [92], it is worth studying if and to what extent the algorithm used for the analytical environment can be deployed on the finite difference environment. The cumulative reward of 64.5 is obtained in this environment if the support pressure is predicted with the algorithm trained in the analytical environment. This result is compared by retraining the model with $\varepsilon_0 = 1$ in the finite difference environment. As it takes approximately 30 min to complete one episode, this training is computationally costly. Therefore, the model is trained only for 50 episodes or about 90% of the expected peak cumulative reward according to Figure 10.

The cumulative reward obtained at each episode is shown in Figure 13. For the first 20 episodes, the cumulative reward oscillates across approximately 65, corresponding to the reward obtained by using the model trained in the analytical environment. The cumulative reward oscillates across 75 between episodes 20 and 37 and stabilises at about 90 from episode 40.

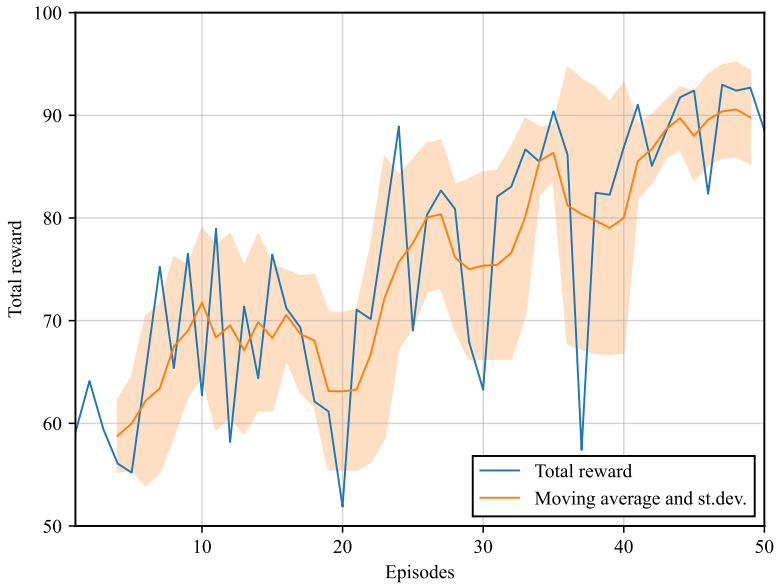


Figure 13. Rewards of the finite difference environment. Cumulative reward vs. no. of episodes; moving average and interval of range ±one standard deviation for the last ten episodes (orange band).

Figure 14 shows the support pressures chosen along the chainage at each episode. This visualisation shows how the initial randomness begins to vanish at episode 20 where the first patterns start to emerge. In particular, the agent learns that the optimal support pressure between chainage 65 and 100 m is 250 kPa. It is also interesting that, between episodes 20 and 37, the agent preferably chooses a support pressure of 150 kPa between chainage 0 and 65 m. Starting from episode 37, it is evident that the optimal support pressure for the first 65 m is 100 kPa. This is consistent with Figure 13, where the cumulative reward starts increasing after episode 20 and increases again after episode 37.

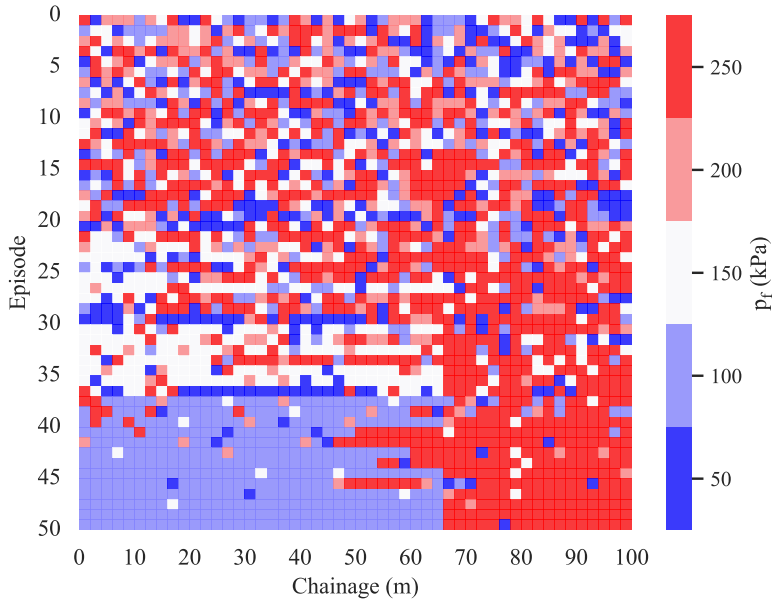


Figure 14. Support pressure chosen by the agent along the chainage for every episode.

Figure 15 shows the settlement after the last episode is completed. The surface settlement is up to 1 cm in the first 65 m of excavation and up to 2 cm in the last 35 m. Hence, it is evident that the actions chosen by the agent keep the settlement within reasonable limits.

From the previous analysis, it seems that the model developed in the analytical environment can be transferred to the finite difference environment provided that the model is retrained. These results are discussed in the next section.

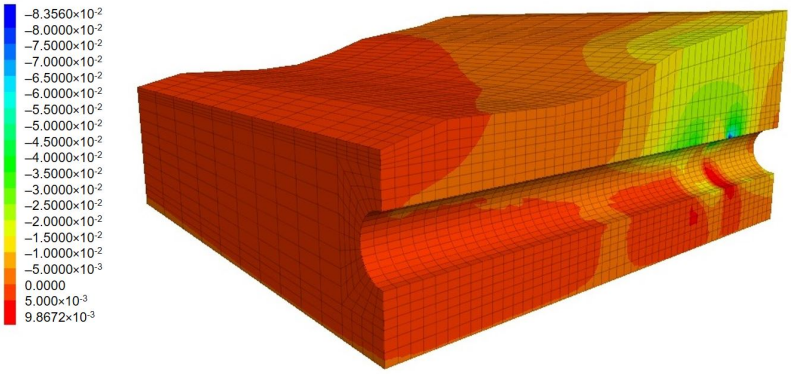


Figure 15. Computed settlement at the last episode (in metres).

4. Discussion

The results show that our model can optimise the support pressure by simultaneously controlling the surface settlement within a reasonable threshold in both analytical and finite difference environments. Implementing model training in an analytical environment is relatively simple and a large number of episodes can be completed fairly fast. Moreover, this class of environments allows for hyperparameter tuning. On the other hand, reinforcement learning training in the finite difference environment (or, more generically, in numerical environments) is rather costly, see also [25]. Therefore, transfer learning is employed for hyperparameter tuning. As shown in the previous section, the model architecture and hyperparameters can be generally transferred to the finite difference environment, on the condition that retraining is performed starting from $\varepsilon_0 = 1.0$.

Some limitations of this study can be highlighted and a strategy to amend them is outlined in the following. Firstly, albeit its use in engineering design, the finite difference

environment cannot completely match reality. This is especially true in light of the simplifications considered in this study, such as the linear-elastic material law with Mohr–Coulomb failure criterion, the simulation of the tunnel lining as a zero displacement boundary condition, the absence of the ring gap and mortar, and the deterministic soil property values. These limitations can be overcome as follows:

1. The adoption of more advanced constitutive models, the simulation of the lining with shell elements, and the simulation of the ring gap and mortar [93,94]. It is perhaps worth noting that different types of segments (in terms of concrete class and reinforcement) and ring gap mortar pressures are chosen in practice. Hence, two additional agents could be implemented to predict the segment types and mortar pressures.
2. The consideration of the spatial variability of soil properties with random fields, by varying the soil properties according to certain statistical distributions and correlation lengths [95]. Since random fields further complicate the environment, more advanced reinforcement learning algorithms might be adopted, such as the 51-atom agent (C51) [96]. Moreover, the definition of the state variables can be improved, e.g., by considering the soil properties at more than one point at each epoch.

The results match the expectation that the agent can be trained to predict the tunnel face support pressure. However, it is striking that the agent does not appear to need any additional training when deployed on random geologies (Section 3.2). This feature could be also theoretically tested with the finite difference environment. However, hyperparameter tuning in this environment is still computationally costly.

The results show that the DQN algorithm can be successfully used to control the tunnel support pressure and adapt to changes in the soil properties, such as variations in unit weight, cohesion, friction angle, and Young's modulus. One added value of the DQN in this context is that it can be used to develop more efficient and effective control strategies for maintaining tunnel face stability compared to traditional methods. The DQN has the capability to generalise to new situations, which can be useful in the case of changes in soil properties or overburden height. Furthermore, the DQN algorithm allows for efficient use of the available data as it is not heavily dependent on its quality, which is a common problem with traditional methods.

5. Conclusions

In this study, the deep Q-network reinforcement learning algorithm was applied to control the tunnel face support pressure during excavation. The algorithm was tested against analytical as well as numerical environments. The analytical environment was used for hyperparameter tuning. The optimised model was used in the numerical environment.

It was found that:

1. The algorithm is capable of predicting the tunnel face support pressure that ensures stability and minimise settlements among a prescribed range of pressures. The algorithm can adapt to geological (soil properties) or geometrical (overburden) changes.
2. An analytical environment is used to optimise the algorithm. The optimal hyperparameters are found as $\Gamma = 0.15$ (discount factor), $\lambda_r = 10^{-3}$ (learning rate), $f = 5$ (synchronisation frequency), $s_m = 10$ (memory size) and $s_b = 2$ (batch size). These hyperparameter values are effective also in the numerical environment.
3. Although the algorithm is trained in a static environment with constant geology, it is also effective with random geological settings. In particular, it is found that using the algorithm trained with constant geology can be used for random geologies without retraining.
4. The maximum cumulative reward plateaus after 400 training episodes and about 90% of the peak performance is reached after 50 episodes.
5. The algorithm proves effective both in the analytical and in the more realistic numerical environment. Training is more computationally costly in the numerical environment. However, the hyperparameter values optimised in the analytical environment can be efficiently adopted.

Future research studies can consider more refined environments (in terms of constitutive models, simulation of the lining, ring gap, mortar, and random fields), provide more advanced state definitions (by considering the soil property values of various points), and use more refined reinforcement learning algorithms.

In spite of some limitations of this method, this study shows that the tunnel face support pressure can be estimated by an intelligent agent for design and possibly even during building operations. In regard to this, a roadmap for field validation can be envisioned. First, the neural network can be pre-trained with monitoring and operational data. Secondly, a scaled model can be constructed and tests conducted either at 1g or in a geotechnical centrifuge. Thirdly, a pilot project with a small TBM, such as those used in microtunnelling, can be carried out.

Author Contributions: Conceptualization, E.S.; methodology, E.S.; software, E.S.; validation, E.S. and C.G.; formal analysis, E.S. and C.G.; investigation, E.S.; resources, W.W.; data curation, E.S. and C.G.; writing—original draft preparation, E.S.; writing—review and editing, C.G. and W.W.; visualization, E.S.; supervision, W.W.; project administration, W.W.; funding acquisition, W.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Otto Pregl Foundation for geotechnical fundamental research.

Data Availability Statement: The data presented in this study are available at https://github.com/soranz84/220721_TBM_RL accessed on 9 March 2023.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AI	artificial intelligence
DQN	deep Q-network
EPB	Earth pressure balance shield
FDM	finite difference method
NATM	New Austrian Tunnelling Method
SPB	slurry pressure balance shield
TBM	tunnel boring machine

List of symbols

A	Cross-sectional area of the tunnel	(m²)
A_i	Action taken at state i	
C	Soil cover	(m)
D	Tunnel diameter	(m)
E	Soil Young's modulus	(MPa)
E_max	Maximum value of the soil Young's modulus	(MPa)
G	Self weight of the sliding wedge	(kN)
K	Experience factor	
N	Number of episodes	
P_V	Vertical load from the soil prism	(kN)
$Q(s,a)$	Value function	
$\hat{Q}(s,a)$	Target value function	
$Q_\mathrm{max}(S_{i+1},a)$	Maximum reward in state S_{i+1} for actions a	
R_i	Reward at state i	
S_i	State of the environment	
T	Shear force on the vertical slip surface	(kN)
a	Action vector	
c	Soil cohesion	(kPa)
c_max	Maximum value of the soil cohesion	(kPa)

f	Synchronisation frequency	
j	Episode counter	
n_a	Random action	
p_f	Tunnel face support pressure	(kPa)
$p_{f,req}$	Tunnel face support pressure required for stability	(kPa)
r	Rewards vector	
s	State vector	
s_b	Batch size	
s_m	Memory size	
t	State counter	
u	Soil settlement above the tunnel	(mm)
x	Tunnel chainage	(m)
Γ	Discount factor	
$\Delta u_{t,j}$	Settlement difference between excavation steps	(mm)
ε	Probability of a random action	
ε_0	Initial probability of a random action	
γ	Soil unit weight	(kN/m³)
γ_{max}	Maximum value of the soil unit weight	(kN/m³)
γ_{sm}	Unit weight of the support medium	(kN/m³)
λ	Stress release due to tunnel construction	
λ_r	Learning rate	
φ	Soil friction angle	(°)
φ_{max}	Maximum value of the soil friction angle	(°)
θ_Q	Parameters of the value function	
θ_T	Parameters of the target value function	
ϑ	Sliding angle	(°)

References

1. Davis, E.H.; Gunn, M.J.; Mair, R.J.; Seneviratine, H.N. The stability of shallow tunnels and underground openings in cohesive material. *Géotechnique* **1980**, *30*, 397–416. [CrossRef]
2. Leca, E.; Dormieux, L. Upper and lower bound solutions for the face stability of shallow circular tunnels in frictional material. *Géotechnique* **1990**, *40*, 581–606. [CrossRef]
3. Anagnostou, G.; Kovári, K. The face stability of slurry-shield-driven tunnels. *Tunn. Undergr. Space Technol.* **1994**, *9*, 165–174. [CrossRef]
4. Anagnostou, G.; Kovári, K. Face stability conditions with earth-pressure-balanced shields. *Tunn. Undergr. Space Technol.* **1996**, *11*, 165–173. [CrossRef]
5. Horn, N. Horizontal ground pressure on vertical faces of tunnel tubes. In Proceedings of the Landeskonferenz der Ungarischen Tiefbauindustrie, Budapest, Hungary, 18–21 June 1961.
6. DAUB. Recommendations for face support pressure calculations for shield tunnelling in soft ground. Guidelines, Deutscher Ausschuss für unterirdisches Bauen, Cologne, Germany, 2016. Available online: bit.ly/3YEMGm2 (accessed on 11 March 2023).
7. Alagha, A.S.; Chapman, D.N. Numerical modelling of tunnel face stability in homogeneous and layered soft ground. *Tunn. Undergr. Space Technol.* **2019**, *94*, 103096. [CrossRef]
8. Sterpi, D.; Cividini, A. A Physical and Numerical Investigation on the Stability of Shallow Tunnels in Strain Softening Media. *Rock Mech. Rock Eng.* **2004**, *37*, 277–298. [CrossRef]
9. Vermeer, P.; Ruse, N.; Dolatimehr, A. Tunnel Heading Stability in Drained Ground. *Felsbau* **2002**, *20*, 8–18.
10. Augarde, C.E.; Lyamin, A.V.; Sloan, S.W. Stability of an undrained plane strain heading revisited. *Comput. Geotech.* **2003**, *30*, 419–430. [CrossRef]
11. Ahmed, M.; Iskander, M. Evaluation of tunnel face stability by transparent soil models. *Tunn. Undergr. Space Technol.* **2012**, *27*, 101–110. [CrossRef]
12. Chen, R.P.; Li, J.; Kong, L.G.; Tang, L.J. Experimental study on face instability of shield tunnel in sand. *Tunn. Undergr. Space Technol.* **2013**, *33*, 12–21. [CrossRef]
13. Kirsch, A. Experimental investigation of the face stability of shallow tunnels in sand. *Acta Geotech.* **2010**, *5*, 43–62. [CrossRef]
14. Lü, X.; Zeng, S.; Zhao, Y.; Huang, M.; Ma, S.; Zhang, Z. Physical model tests and discrete element simulation of shield tunnel face stability in anisotropic granular media. *Acta Geotech.* **2020**, *15*, 3017–3026. [CrossRef]
15. Lü, X.; Zhou, Y.; Huang, M.; Zeng, S. Experimental study of the face stability of shield tunnel in sands under seepage condition. *Tunn. Undergr. Space Technol.* **2018**, *74*, 195–205. [CrossRef]
16. Chambon, P.; Corté, J. Shallow Tunnels in Cohesionless Soil: Stability of Tunnel Face. *J. Geotech. Eng.* **1994**, *120*, 1148–1165. [CrossRef]

17. Mair, R. Centrifugal Modelling of Tunnel Construction in Soft Clay. Ph.D. Thesis, University of Cambridge, Cambridge, UK, 1979.
18. Soranzo, E.; Wu, W., Centrifuge Test of Face Stability of Shallow Tunnels in Unsaturated Soil. In *Poromechanics V*; American Society of Civil Engineers: Reston, VA, USA, 2013; pp. 1326–1335. [CrossRef]
19. Jong, S.; Ong, D.; Oh, E. State-of-the-art review of geotechnical-driven artificial intelligence techniques in underground soil-structure interaction. *Tunn. Undergr. Space Technol.* **2021**, *113*, 103946. [CrossRef]
20. Ebid, A. 35 Years of (AI) in Geotechnical Engineering: State of the Art. *Geotech. Geol. Eng.* **2021**, *39*, 637–690. [CrossRef]
21. Zhang, W.; Zhang, R.; Wu, C.; Goh, A.T.C.; Lacasse, S.; Liu, Z.; Liu, H. State-of-the-art review of soft computing applications in underground excavations. *Geosci. Front.* **2020**, *11*, 1095–1106. [CrossRef]
22. Marcher, T.; Erharter, G.; Winkler, M. Machine Learning in tunnelling—Capabilities and challenges. *Geomech. Tunn.* **2020**, *13*, 191–198. [CrossRef]
23. Shahrour, I.; Zhang, W. Use of soft computing techniques for tunneling optimization of tunnel boring machines. *Undergr. Space* **2021**, *6*, 233–239. [CrossRef]
24. Soranzo, E.; Guardiani, C.; Wu, W. A soft computing approach to tunnel face stability in a probabilistic framework. *Acta Geotech.* **2022**, *17*, 1219–1238. [CrossRef]
25. Soranzo, E.; Guardiani, C.; Wu, W. The application of reinforcement learning to NATM tunnel design. *Undergr. Space* **2022**, in press. [CrossRef]
26. Qin, S.; Xu, T.; Zhou, W.H. Predicting Pore-Water Pressure in Front of a TBM Using a Deep Learning Approach. *Int. J. Geomech.* **2021**, *21*, 04021140. [CrossRef]
27. Kim, D.; Pham, K.; Oh, J.Y.; Lee, S.J.; Choi, H. Classification of surface settlement levels induced by TBM driving in urban areas using random forest with data-driven feature selection. *Autom. Constr.* **2022**, *135*, 104109. [CrossRef]
28. Kim, D.; Kwon, K.; Pham, K.; Oh, J.Y.; Choi, H. Surface settlement prediction for urban tunneling using machine learning algorithms with Bayesian optimization. *Autom. Constr.* **2022**, *140*, 104331. [CrossRef]
29. Lee, H.K.; Song, M.K.; Lee, S.S. Prediction of Subsidence during TBM Operation in Mixed-Face Ground Conditions from Realtime Monitoring Data. *Appl. Sci.* **2021**, *11*, 12130. [CrossRef]
30. Erharter, G.H.; Hansen, T.F. Towards optimized TBM cutter changing policies with reinforcement learning. *Geomech. Tunn.* **2022**, *15*, 665–670. [CrossRef]
31. Liu, Y.; Huang, S.; Wang, D.; Zhu, G.; Zhang, D. Prediction Model of Tunnel Boring Machine Disc Cutter Replacement Using Kernel Support Vector Machine. *Appl. Sci.* **2022**, *12*, 2267. [CrossRef]
32. Mahmoodzadeh, A.; Mohammadi, M.; Hashim Ibrahim, H.; Nariman Abdulhamid, S.; Farid Hama Ali, H.; Mohammed Hasan, A.; Khishe, M.; Mahmud, H. Machine learning forecasting models of disc cutters life of tunnel boring machine. *Autom. Constr.* **2021**, *128*, 103779. [CrossRef]
33. Hou, S.; Liu, Y.; Zhuang, W.; Zhang, K.; Zhang, R.; Yang, Q. Prediction of shield jamming risk for double-shield TBM tunnels based on numerical samples and random forest classifier. *Acta Geotech.* **2022**, *18*, 495–517. [CrossRef]
34. Lin, P.; Xiong, Y.; Xu, Z.; Wang, W.; Shao, R. Risk assessment of TBM jamming based on Bayesian networks. *Bull. Eng. Geol. Environ.* **2021**, *81*, 47. [CrossRef]
35. Liu, M.; Liao, S.; Yang, Y.; Men, Y.; He, J.; Huang, Y. Tunnel boring machine vibration-based deep learning for the ground identification of working faces. *J. Rock Mech. Geotech. Eng.* **2021**, *13*, 1340–1357. [CrossRef]
36. Wellmann, F.; Amann, F.; de la Varga, M.; Chudalla, N. Automated geological model updates during TBM operation—An approach based on probabilistic machine learning concepts. *Geomech. Tunn.* **2022**, *15*, 635–641. [CrossRef]
37. Erharter, G.H.; Marcher, T. On the pointlessness of machine learning based time delayed prediction of TBM operational data. *Autom. Constr.* **2021**, *121*, 103443. [CrossRef]
38. Sheil, B. Discussion of "on the pointlessness of machine learning based time delayed prediction of TBM operational data" by Georg H. Erharter and Thomas Marcher. *Autom. Constr.* **2021**, *124*, 103559. [CrossRef]
39. Bai, X.D.; Cheng, W.C.; Li, G. A comparative study of different machine learning algorithms in predicting EPB shield behaviour: a case study at the Xi'an metro, China. *Acta Geotech.* **2021**, *16*, 4061–4080. [CrossRef]
40. Guo, D.; Li, J.; Jiang, S.H.; Li, X.; Chen, Z. Intelligent assistant driving method for tunnel boring machine based on big data. *Acta Geotech.* **2022**, *17*, 1019–1030. [CrossRef]
41. Zhang, N.; Zhang, N.; Zheng, Q.; Xu, Y.S. Real-time prediction of shield moving trajectory during tunnelling using GRU deep neural network. *Acta Geotech.* **2022**, *17*, 1167–1182. [CrossRef]
42. Benardos, A.; Kaliampakos, D. Modelling TBM performance with artificial neural networks. *Tunn. Undergr. Space Technol.* **2004**, *19*, 597–605. [CrossRef]
43. Feng, S.; Chen, Z.; Luo, H.; Wang, S.; Zhao, Y.; Liu, L.; Ling, D.; Jing, L. Tunnel boring machines (TBM) performance prediction: A case study using big data and deep learning. *Tunn. Undergr. Space Technol.* **2021**, *110*, 103636. [CrossRef]
44. Gao, B.; Wang, R.; Lin, C.; Guo, X.; Liu, B.; Zhang, W. TBM penetration rate prediction based on the long short-term memory neural network. *Undergr. Space* **2021**, *6*, 718–731. [CrossRef]
45. Xu, H.; Zhou, J.; Asteris, P.G.; Jahed Armaghani, D.; Tahir, M.M. Supervised Machine Learning Techniques to the Prediction of Tunnel Boring Machine Penetration Rate. *Appl. Sci.* **2019**, *9*, 3715. [CrossRef]
46. Li, J.; Li, P.; Guo, D.; Li, X.; Chen, Z. Advanced prediction of tunnel boring machine performance based on big data. *Geosci. Front.* **2021**, *12*, 331–338. [CrossRef]

47. Mahmoodzadeh, A.; Nejati, H.R.; Mohammadi, M.; Hashim Ibrahim, H.; Rashidi, S.; Ahmed Rashid, T. Forecasting tunnel boring machine penetration rate using LSTM deep neural network optimized by grey wolf optimization algorithm. *Expert Syst. Appl.* **2022**, *209*, 118303. [CrossRef]
48. International Business Machines. Supervised Learning, 2020. Available online: https://www.ibm.com/cloud/learn/supervised-learning (accessed on 11 March 2023).
49. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [CrossRef] [PubMed]
50. Schrittwieser, J.; Antonoglou, I.; Hubert, T.; Simonyan, K.; Sifre, L.; Schmitt, S.; Guez, A.; Lockhart, E.; Hassabis, D. Graepel, T.; Lillicrap, T.; et al. Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature* **2020**, *588*, 604–612. [CrossRef] [PubMed]
51. Silver, D.; Huang, A.; Maddison, C.; Guez, A.; Sifre, L.; van den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; et al. Mastering the game of Go with deep neural networks and tree search. *Nature* **2016**, *529*, 484–503. [CrossRef]
52. Silver, D.; Schrittwieser, J.; Simonyan, K.; Antonoglou, I.; Huang, A.; Guez, A.; Hubert, T.; Baker, L.L.; Lai, M.; Bolton, A.; et al. Mastering the game of Go without human knowledge. *Nature* **2017**, *550*, 354–370. [CrossRef]
53. Silver, D.; Hubert, T.; Schrittwieser, J.; Antonoglou, I.; Lai, M.; Guez, A.; Lanctot, M.; Sifre, L.; Kumaran, D.; Graepel, T.; et al. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science* **2018**, *362*, 1140–1144. [CrossRef] [PubMed]
54. Shahrabi, J.; Adibi, M.; Mahootchi, M. Computers and Industrial Engineering. *Nature* **2017**, *110*, 75–82. [CrossRef]
55. Ipek, E.; Mutlu, O.; Martínez, J.F.; Caruana, R. Self-Optimizing Memory Controllers: A Reinforcement Learning Approach. In Proceedings of the 2008 International Symposium on Computer Architecture, Virtual, 21–25 June 2008; pp. 39–50. [CrossRef]
56. Martinez, J.F.; Ipek, E. Dynamic Multicore Resource Management: A Machine Learning Approach. *IEEE Micro* **2009**, *29*, 8–17. [CrossRef]
57. Li, L.; Chu, W.; Langford, J.; Schapire, R.E. A Contextual-Bandit Approach to Personalized News Article Recommendation. *arXiv* **2010**, arXiv:1003.0146.
58. Theocharous, G.; Thomas, P.S.; Ghavamzadeh, M. Personalized Ad Recommendation Systems for Life-Time Value Optimization with Guarantees. In Proceedings of the 24th International Conference on Artificial Intelligence, Buenos Aires, Argentina, 25–31 July 2015; pp. 1806–1812.
59. Kiran, B.R.; Sobh, I.; Talpaert, V.; Mannion, P.; Sallab, A.A.A.; Yogamani, S.K.; Pérez, P. Deep Reinforcement Learning for Autonomous Driving: A Survey. *arXiv* **2020**, arXiv:2002.00444.
60. Gao, X. Deep reinforcement learning for time series: Playing idealized trading games. *arXiv* **2018**, arXiv:1803.03916.
61. Paulus, R.; Xiong, C.; Socher, R. A Deep Reinforced Model for Abstractive Summarization. *arXiv* **2017**, arXiv:1705.04304.
62. Yu, C.; Liu, J.; Nemati, S. Reinforcement Learning in Healthcare: A Survey. *arXiv* **2019**, arXiv:1908.08796.
63. Mousavi, S.; Schukat, M.; Howley, E. Deep Reinforcement Learning: An Overview. In Proceedings of the SAI Intelligent Systems Conference (IntelliSys) 2016, London, UK, 21–22 September 2016; pp. 426–440. ._32. [CrossRef]
64. Erharter, G.; Marcher, T.; Hansen, T.; Liu, Z. Reinforcement learning based process optimization and strategy development in conventional tunnelling. *Autom. Constr.* **2021**, *127*, 103701. [CrossRef]
65. Zhang, P.; Li, H.; Ha, Q.; Yin, Z.Y.; Chen, R.P. Reinforcement learning based optimizer for improvement of predicting tunneling-induced ground responses. *Adv. Eng. Informatics* **2020**, *45*, 101097. [CrossRef]
66. Van Rossum, G.; Drake, F.L. *Python 3 Reference Manual*; CreateSpace: Scotts Valley, CA, USA, 2009.
67. Andrew, A. Reinforcement Learning: An Introduction. *Kybernetes* **1998**, *27*, 1093–1096. [CrossRef]
68. Bowyer, C. Characteristics of Rewards in Reinforcement Learning, 2022. Available online: https://medium.com/mlearning-ai/characteristics-of-rewards-in-reinforcement-learning-f5722079aef5 (accessed on 11 March 2023).
69. Singh, N. A Comprehensive Guide to Reinforcement Learning. Available online: https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-to-reinforcement-learning (accessed on 18 May 2022).
70. Gütter, W.; Jäger, M.; Rudigier, G.; Weber, W. TBM versus NATM from the contractor's point of view. *Geomech. Tunn.* **2011**, *4*, 327–336. [CrossRef]
71. Schuck, N.W.; Wilson, R.; Niv, Y. Chapter 12 - A State Representation for Reinforcement Learning and Decision-Making in the Orbitofrontal Cortex. In *Goal-Directed Decision Making*; Morris, R., Bornstein, A., Shenhav, A., Eds.; Academic Press: New York, NY, USA, 2018; pp. 259–278. [CrossRef]
72. Kit Machine. Is Domain Knowledge Important for Machine Learning? 2022. Available online: https://www.kit-machines.com/domain-knowledge-machine-learning/ (accessed on 11 March 2023).
73. Chen, R.; Tang, L.; Ling, D.; Chen, Y. Face stability analysis of shallow shield tunnels in dry sandy ground using the discrete element method. *Comput. Geotech.* **2011**, *38*, 187–195. [CrossRef]
74. Leca, E.; New, B. Settlements induced by tunneling in Soft Ground. *Tunn. Undergr. Space Technol.* **2007**, *22*, 119–149. [CrossRef]
75. ITAtech Activity Group Investigation. *Geophysical Ahead Investigation Methods Seismic Methods*; Technical Report ITAtech Report No. 10; International Tunnelling Association: Salt Lake City, UT, USA, 2018.
76. Shirlaw, N. Setting operating pressures for TBM tunnelling. In *Proceedings of the HKIE Geotechnical Division Annual Seminar, 2012: Geotechnical Aspects of Tunnelling for Infrastructure, Hong Kong, China, 13 May 2022*; The Hong Kong Institution of Engineers: Hong Kong, China, 2012; pp. 7–28.

77. Virtanen, P.; Gommers, R.; Oliphant, T.E.; Haberland, M.; Reddy, T.; Cournapeau, D.; Burovski, E.; Peterson, P.; Weckesser, W.; Bright, J.; et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nat. Methods* **2020**, *17*, 261–272. [CrossRef] [PubMed]
78. Li, Y. Deep Reinforcement Learning: An Overview. *arXiv* **2017**, arXiv:1701.07274.
79. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; Riedmiller, M. Playing Atari with Deep Reinforcement Learning. *arXiv* **2013.** arXiv:1312.5602
80. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al., PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*; Curran Associates: San Diego, CA, USA, 2019; Volume 32, pp. 8024–8035.
81. Zai, A.; Brown, B. *Deep Reinforcement Learning in Action*; Manning: New York, NY, USA, 2020.
82. Bengio, Y., Practical Recommendations for Gradient-Based Training of Deep Architectures. In *Neural Networks: Tricks of the Trade: Second Edition*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 437–478. [CrossRef]
83. Soranzo, E.; Guardiani, C.; Saif, A.; Wu, W. A Reinforcement Learning approach to the location of the non-circular critical slip surface of slopes. *Comput. Geosci.* **2022**, *166*, 105182. [CrossRef]
84. Lin, L. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Mach. Learn.* **1992**, *8*, 293–321. [CrossRef]
85. McCloskey, M.; Cohen, N.J. Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. In *Psychology of Learning and Motivation*; Bower, G.H., Ed.; Academic Press: Cambridge, MA, USA, 1989; Volume 24, pp. 109–165. [CrossRef]
86. Alsahly, A.; Stascheit, J.; Meschke, G. Advanced finite element modeling of excavation and advancement processes in mechanized tunneling. *Adv. Eng. Softw.* **2016**, *100*, 198–214. . [CrossRef]
87. Demagh, R.; Emeriault, F. 3D Modelling of Tunnel Excavation Using Pressurized Tunnel Boring Machine in Overconsolidated Soils. *Stud. Geotech. Mech.* **2014**, *35*, 3–17. [CrossRef]
88. Hasanpour, R. Advance numerical simulation of tunneling by using a double shield TBM. *Comput. Geotech.* **2014**, *57*, 37–52. [CrossRef]
89. Hasanpour, R.; Rostami, J.; Ünver, B. 3D finite difference model for simulation of double shield TBM tunneling in squeezing grounds. *Tunn. Undergr. Space Technol.* **2014**, *40*, 109–126. [CrossRef]
90. Itasca Consulting Group. *FLAC3D (Fast Lagrangian Analysis of Continua)*; Itasca Consulting Group: Minneapolis, MN, USA, 2009.
91. Dias, T.G.S.; Bezuijen, A. TBM Pressure Models—Observations, Theory and Practice. In *Proceedings of the Volume 5: Geotechnical Synergy in Buenos Aires 2015, Argentina, 5–18 November 2015*; Sfriso, A.O., Ed.; IOS Press: Clifton, NJ, USA, 2015; pp. 347–375. [CrossRef]
92. Vilalta, R.; Giraud-Carrier, C.; Brazdil, P.; Soares, C. Inductive Transfer. In *Encyclopedia of Machine Learning*; Springer: Boston, MA, USA, 2010; pp. 545–548. [CrossRef]
93. Kasper, T.; Meschke, G. A 3D finite element simulation model for TBM tunnelling in soft ground. *Int. J. Numer. Anal. Methods Geomech.* **2004**, *28*, 1441–1460. [CrossRef]
94. Kasper, T.; Meschke, G. On the influence of face pressure, grouting pressure and TBM design in soft ground tunnelling. *Tunn. Undergr. Space Technol.* **2006**, *21*, 160–171. [CrossRef]
95. Fenton, G.; Griffiths, D. *Risk Assessment in Geotechnical Engineering*; John Wiley and Sonds: Hoboken, NJ, USA, 2008. [CrossRef]
96. Bellemare, M.G.; Dabney, W.; Munos, R. A Distributional Perspective on Reinforcement Learning. *arXiv* **2017**, arXiv:1707.06887.