Published in partnership with the Shanghai Institute of Ceramics of the Chinese Academy of Sciences

https://doi.org/10.1038/s41524-025-01731-1

Acceleration of crystal structure relaxation with deep reinforcement learning

Check for updates

Elena Trukhan ✉, Efim Mazhnik & Artem R. Oganov

We introduce a Deep Reinforcement Learning (DRL) model for crystal structure relaxation and compare different types of neural network architectures and reinforcement learning algorithms for this purpose. Numerical experiments are conducted on Al-Fe structures, with potential energy surfaces generated using EAM potentials. We examine the influence of parameter settings on model performance and benchmark the best-performing models against classical optimization algorithms. Additionally, the model's capacity to generalize learned interaction patterns from smaller atomic systems to more complex systems is assessed. The results demonstrate the potential of DRL models to enhance the efficiency of structure relaxation compared to classical optimizers.

Structure relaxation is the optimization of the positions of atoms and the lattice geometry in the process of searching for the minimum of the potential energy and interatomic forces, which corresponds to a (meta)stable state. It is a fundamental part of atomistic and molecular modeling and is used in a wide range of scientific and engineering tasks to study the properties and behavior of various materials and systems.

Many structure relaxation algorithms formalize the problem as a function minimization task and search for a (meta)stable state iteratively: at each step, the potential energy, its gradients (forces), and in some methods, the Hessian matrix, are evaluated using various techniques, including first-principles methods like the density functional theory (DFT) and the many-body perturbation theory (MBPT), as well as empirical potentials, machine learning potentials, and others. The data obtained from these calculations are then used in optimization algorithms such as steepest descent[1], conjugate gradients[2], Monte Carlo[3,4], and molecular dynamics[5]. These algorithms guide the shift of atoms toward the direction of minimum energy.

The issue arises when the complexity of the structure increases, such as with the growth in the number of atoms N in a unit cell or the diversity of chemical elements. This leads to an increase in the time required for relaxation, because the computational complexity of calculating the energy and forces increases with N, starting from $O(N^3)$ for DFT[6] up to $O(N^7)$[7,8] and reaching exponential complexity $O(e^{2N})$ with more accurate methods. The complexity of the potential energy surface (PES) also increases with system size, resulting in a greater number of local minima that the relaxation process must navigate[9].

Deep Reinforcement Learning (DRL) can be used in this case to predict the most efficient steps of optimization and to overcome the mentioned time problem by reducing the number of steps. In the field of materials science, Reinforcement Learning has been used mainly for the design of new materials[10–13], synthesis planning[14] and microstructure optimization[15] but relaxation based on RL is promising for several reasons:

- It enables the utilization of experience accumulated over a large number of optimizations, which are ignored by existing optimization methods. This is particularly beneficial because many tasks involve the relaxation of structures that have already been optimized, allowing us to leverage known paths from disturbed to relaxed structures. For example, studying the catalytic properties of a material involves the relaxation of structures that differ only in the position of several molecules on the surface.
- RL addresses the challenge of generating representative datasets, as agent gathers data directly by interacting with the environment.
- The objective of reducing the number of relaxation steps can be established both directly by selecting the reward function as a function of the number of steps and indirectly through the introduction of a discount parameter.

In this work, we introduce a Deep Reinforcement Learning (DRL) model for the fixed unit cell crystal structure relaxation process. Previous studies have applied reinforcement learning (RL) with structure representations based on local atomic environment encodings for molecular geometry optimization[16], predicting minimum-energy reaction pathways[17] and global energy minimization[18]. However, to our knowledge, this is the first work that explicitly focuses on the number of optimization steps to a particular local minimum of crystal structure—a key factor in reducing computational costs when using expensive interatomic potentials.

Crystal structures are represented as crystal graphs, following the methodology of Xie et al. (2018)[19]. Two types of neural networks are tested in this work: Crystal Graph Convolutional Neural Networks (CGCNN)[19] and

Material Discovery Laboratory, Skolkovo Institute of Science and Technology, Moscow, Russia. ✉e-mail: Elena.Trukhan@skoltech.ru

E(3)-Equivariant Tensor Field Networks[20]. CGCNN is a conventional model for crystal systems that naturally handles periodic connectivity, capturing local atomic environments through graph convolutions. It has been applied for material property prediction and serves as a baseline for other graph neural network (GNN) architectures[19,21–23]. However, CGCNN and similar models are not equivariant with respect to rotations and reflections—a limitation when processing crystal graph features or predicting outputs (such as atomic shifts in our task) that transform as geometric tensors. In such cases, Tensor Field Networks (TFNs) present a principled solution, as they rigorously preserve the transformation properties of tensors under Euclidean symmetries[24]. TFNs have proven particularly successful in applications requiring strict geometric consistency, including molecular dynamics (force prediction)[25,26] and materials property prediction[27–29], where maintaining correct tensor transformation behavior is physically critical. The Twin Delayed Deep Deterministic Policy Gradient (TD3)[30] algorithm is primarily utilized for model training, as it is a standard choice for continuous control tasks and taking into account that it employs deterministic policy, it might be more natural for deterministic environments like structure relaxation[31]. However, for comparison, we also tested algorithm with stochastic policy and selected the Soft Actor-Critic (SAC)[32], because it maximizes both reward and entropy, inherently promoting exploration. Due to this feature, SAC has been reported to outperform TD3 in some high-dimensional tasks[31]. Results for this method are presented in Supplementary Section 7.

We compare different architectures and RL algorithms taking as a benchmark the Al-Fe system described by EAM potentials[33–35] used for the PES generation. First of all, dependence of the model performance on parameter settings is studied. Next, we compare the best models with classical optimizers: Broyden-Fletcher-Goldfarb-Shanno[36–39] and conjugate gradient[40]. Finally, the model's ability to generalize learned patterns of interactions from systems with a small number of atoms/chemical elements to systems with a larger number of atoms/chemical elements is investigated.

Results

Considering that this is, to our knowledge, the first work where reinforcement learning is applied to the crystal structure relaxation task, we find that it is important to provide a detailed description of how different model settings influence the algorithm's performance for further development of this approach. Accordingly, this section is structured as follows: first, we cover the investigation of the method's technical aspects, while subsequent sections demonstrate the algorithm's performance on practical tasks. Additionally, an analysis of the influence of the discount factor and exploration level on model performance is provided in Supplementary Section 8.

The list of structures in all datasets and the corresponding hyperparameters are provided in Supplementary Sections 9–10.

Comparison of TFN and CGCNN models

For the comparison of two suggested architectures, the TD3 Agent was trained on CsCl-type structure of AlFe with 5 different random seeds. Averaged learning curves are presented in Fig. 1.

As can be seen, the CGCNN model is ineffective in performing the task of structure relaxation because it does not reduce the maximum force at the last step and does not increase the full score, as it should be in the case of successful training of RL Agent. This may be attributed to the continuous and high-dimensional nature of the task, both in terms of state and action spaces. From this perspective, the TFN model manages the task of relaxation better for two main reasons: firstly, it does not require training on augmented data to navigate through crystal structures differing only in rotations; secondly, the TFN model predicts actions not from the whole space of all possible atomic shifts, as in the case of CGCNN, but from a subspace of shifts restricted by the symmetry of the structure. It has been proven that any E(3)-equivariant model obeys Curie's principle[27], so the symmetry of the output can only be of equal or higher symmetry than the input. As a result, reducing the action space allows the model to quickly find the optimal policy. The drawback is that this feature limits our model because we cannot guarantee that geodesic path in the action space to the minimum always has higher symmetry than the symmetry of the structure. However, the investigation of this question is beyond the scope of this work.

As a result of the presented comparison, we further tested only TFN models.

Comparison of reward functions and sensitivity issues

The TD3 Agent was trained with different reward functions, outlined in Eqs. (2)–(4) on CsCl-type structure of AlFe. The results are depicted in Fig. 2a, b). We also compare the performance of the best trained models with BFGS on relaxation up to higher values of the force threshold ϵ_1 (see Fig. 2c).

First of all, one can see from Fig. 2c) that our models effectively relax CsCl-type structure of AlFe up to ϵ_1 = 0.1 eV/A; however, reducing the forces further is challenging. This issue may stem from insufficient model sensitivity near the minimum, as the differences in node and edge features of crystal structures approach the numerical error limits of TFN models, and the spatial and angular resolution of the model architecture might be insufficient to distinguish them (see further discussions in Supplementary Section 4). Consequently, adopting a *log force* reward function has led to a reduction in the number of steps required for relaxation, indicating increased sensitivity for at least the reward function in the vicinity of the minimum.

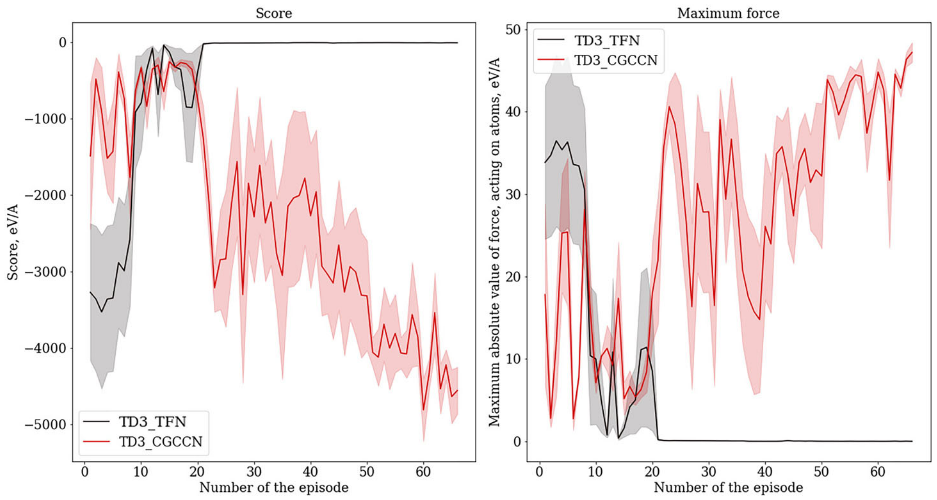

Fig. 1 | Learning curves of TD3 Agents with CGCNN and TFN architectures, trained on CsCl-type structure of AlFe. Curves are averaged over 5 trials with different random seeds, with the shaded area representing half a standard deviation.

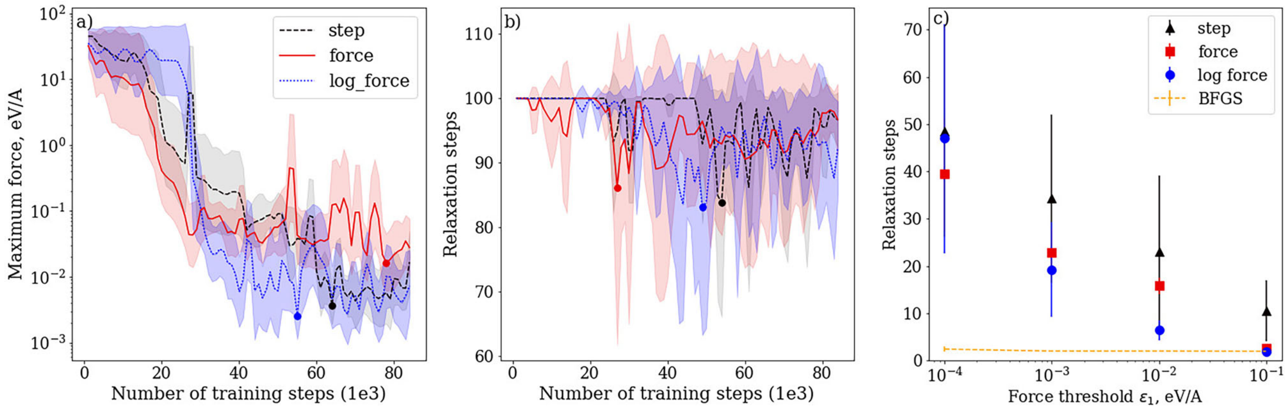

Fig. 2 | Comparison of TD3 Agents trained with different reward functions on CsCl-type AlFe. a, b Learning curves of TD3 Agents with different reward functions, given by Eqs. (2)–(4). Curves are averaged over 5 seeds, with the shaded area representing the standard deviation. Circles mark the lowest average relaxation steps and maximum force at the last step. **c** Performance of the Agents, corresponding to the models with the best results with respect to relaxation steps (corresponds to the circles on (**b**)). The plot shows the median value with an interdecile range of a sample of 50 relaxation episodes.

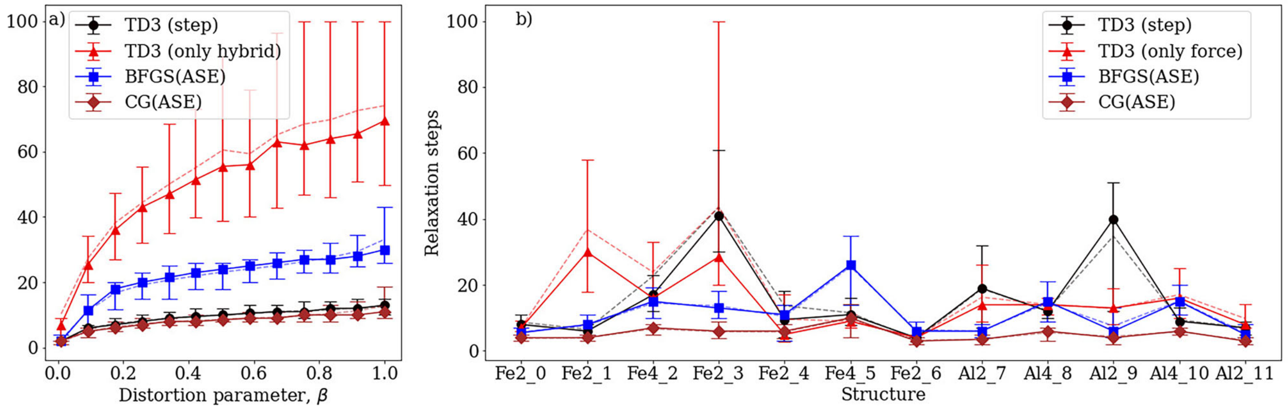

Fig. 3 | Comparison of classical algorithms and TD3 Agents trained with and without switching to the *step* reward after pertaining. a With *hybrid* reward function ($\hat{w}_1 = 1, \hat{w}_2 = 0, \hat{w}_3 = 0.5$) on hypothetical I4/mmm structure of Al (**b**) with *force* reward function on the set of single-element Al and Fe structures (the description of the set is presented in Supplementary Table S2). All models were trained with $\beta = 0.5$. For Al the model is tested for relaxation over a range of distortion parameter values β, displayed on the x-axis. For each point on the plots relaxation was conducted 50 times. The solid line indicates the median value with an interdecile range, while the dashed line represents the mean value.

Secondly, it can be noticed that usage of the *step* reward function does not provide better performance in terms of the number of steps required for relaxation, while from the definition of this reward function in Eq. (3), it should optimize this value directly. This may be explained by a piecewise-constant nature of the action-value function which causes the Agent to adopt a suboptimal policy if the Agent does not face the terminal steps at the beginning of training (for the detailed discussion of this effect see Supplementary Section 3).

To address this issue we tested an approach where the Agent firstly is pretrained with more smooth functions, for example, *hybrid* reward with $\hat{w}_1 = 1, \hat{w}_2 = 0, \hat{w}_3 = 0.5$ and then the reward function is switched to *step* with consistently reducing ϵ_1 in Eq. (6) during training. Each stage of model training continued until the Agent reached a predetermined limit on the number of steps on relaxation.

In Fig. 3, a comparison is presented between the performance of a model trained exclusively on the smooth reward function and one trained on the *step* reward function after pretraining. This comparison is made for two cases: (1) when the Agent is trained on a single structure (the hypothetical I4/mmm structure of Al) and (2) when it is trained on a set of multiple structures simultaneously (a dataset of single-element Al and Fe structures). During training with the *step* reward function, the parameter ϵ_1 was gradually reduced—from 0.1 eV/Å to 0.01 eV/Å for the hypothetical I4/mmm structure of Al, and from 0.1 eV/Å to 0.05 eV/Å for the dataset of single-element Al and Fe structures.

As shown in the Results Section, this approach enables the model to achieve performance comparable to CG when the Agent is trained on a single structure. Furthermore, Fig. 3a) demonstrates the model's ability to extrapolate its experience to higher distortions. Despite being trained with a distortion parameter $\beta = 0.5$, the model successfully relaxes structures with higher β values during testing. However, as illustrated in Fig. 3b), this method does not guarantee improved results across all structures when the model is trained on a diverse dataset of different configurations.

Comparison with classical optimizers

We compare the performance of the algorithms trained on structures with different numbers of atoms per unit cell N using the sequential approach described in the previous section. For $N = 2$ we use results on the CsCl-type structure of AlFe (see Supplementary Fig. S5a), and for $N = 4$—the results for hypothetical I4/mmm structure of Al (see Fig. 3a). For $N = 8, 10, 20$ the structures are generated using USPEX code[41]. For each N the models are trained on a single structure and tested to perform the relaxation of the same structure but randomly distorted at the beginning of the testing episode, as described in the Methods section. In the case of $N = 2–8$ the relaxation during training of the models and further comparison with classical

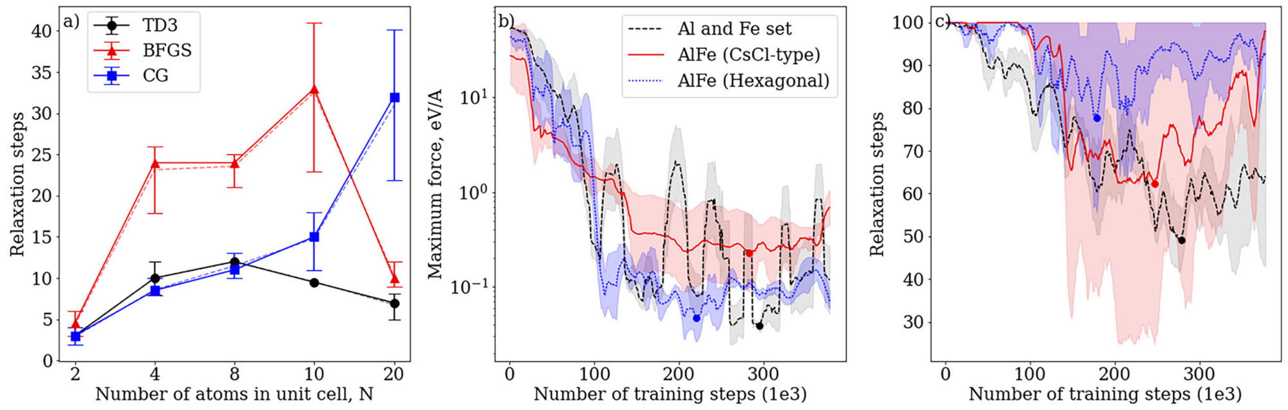

Fig. 4 | A comparison of the model with classical optimizers and learning curves. **a** Comparison of TD3 model and classical optimizers in relaxation of structures with different number of atoms N in the unit cell. For $N = 2-8$ the relaxation was conducted up to $\epsilon_1 = 0.01$ eV/A, while for $N = 10$ and $N = 20$ up to $\epsilon_1 = 0.2$ eV/A and $\epsilon_1 = 0.25$ eV/A correspondingly. For each point on the plot, relaxation was conducted 50 times. The solid line indicates the median value with an interdecile range, while the dashed line represents the mean value. **b**, **c** Learning curve of the TD3 model, trained on the set of single-element Al and Fe structures with *force* reward function (black) and its performance on CsCl-type (red) and simple hexagonal (blue) structures of AlFe during test. The shaded region represents a standard deviation of the average evaluation over 2 trials.

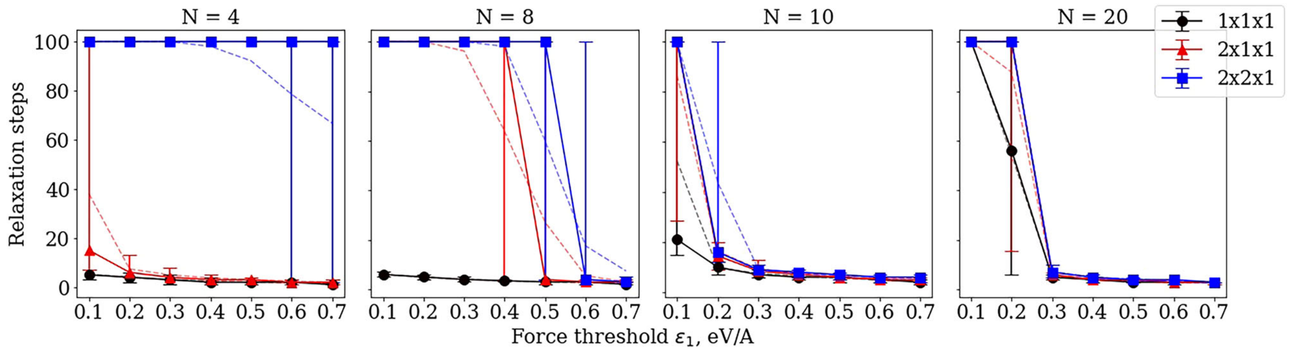

Fig. 5 | Number of steps required for relaxation of supercells by TD3 Agents trained on structures with different number of atoms *N* in the unit cell. Performance is measured for different ϵ_1 values, displayed on the x-axis. The y-axis represents the number of steps taken by the optimizer for the relaxation of the structure, averaged over 50 runs. The solid line indicates the median value with an interdecile range, while the dashed line represents the mean value.It should be noticed that in the case of $N = 20$ the model was trained to perform the relaxation up to $\epsilon_1 = 0.25$ eV/A, and for $N = 10$ up to $\epsilon_1 = 0.2$ eV/A.

algorithms are conducted up to $\epsilon_1 = 0.01$ eV/A, while for $N = 10$ and $N = 20$ it was conducted up to $\epsilon_1 = 0.2$ eV/A and $\epsilon_1 = 0.25$ eV/A, correspondingly due to the sensitivity issues mentioned earlier. As one can see in Fig. 4a), the results of our algorithms are similar to classical optimizers for a small number of atoms N, while for higher values of N it starts to outperform them.

Generalizability of the model from simple to complex structures

An important aspect of the model that we wanted to investigate was its ability to generalize interaction patterns from small systems to more complex ones. From a physical point of view, the ordering of atoms in crystal structures is mainly conditioned by local interactions with the closest neighbors. Therefore, we can expect that an RL Agent based on GCNN will be able to extrapolate its experience from small structures to more complex ones, as the main idea of convolution is to find the correlation between target properties and local interactions. We can expect this ability to be present in two cases:

- The number of atoms per unit cell is increased. In this case, the ability to extrapolate means that the model effectively selects the subspace of atoms that strongly influence the position of the target node in the unit cell and accounts for the fact that, in a large structure distant atoms weakly influence each other.
- The chemical diversity is increased. Increasing chemical diversity leads to new types of interactions that the model needs to account for, but it can generalize these patterns from its experience with other types of interactions.

To investigate the ability of our model to extrapolate to higher chemical diversity, the model was trained on a set of single-element Al and Fe structures. During testing the model performed a relaxation of CsCl-type and simple hexagonal structures of AlFe without being trained on them. The results are presented in Fig. 4b, c).

As one can see, there is a strong correlation between the results for AlFe and the set of single-element structures, indicating that the model is able to generalize Al-Al and Fe-Fe interactions to the case of Al-Fe interactions. The best result for the model, presented in Fig. 4c), in the case of CsCl-type structure of AlFe relaxation is 11.3 ± 7.6 steps, which is close to the best results of the model trained solely on this configuration (see Supplementary Fig. S5a).

To investigate the ability of our model to extrapolate to a larger number of atoms per unit cell, we test how the models trained on single structures with different number of atoms per unit cell relax the corresponding supercells $2 \times 1 \times 1$ and $2 \times 2 \times 1$ without being trained on them. The results are presented in Fig. 5.

As expected, increasing the number of atoms in the structures in the training dataset led to better extrapolation to larger configurations. Consequently, we can anticipate that at some point, the model will be able to relax large structures without being pretrained on them. This is because the

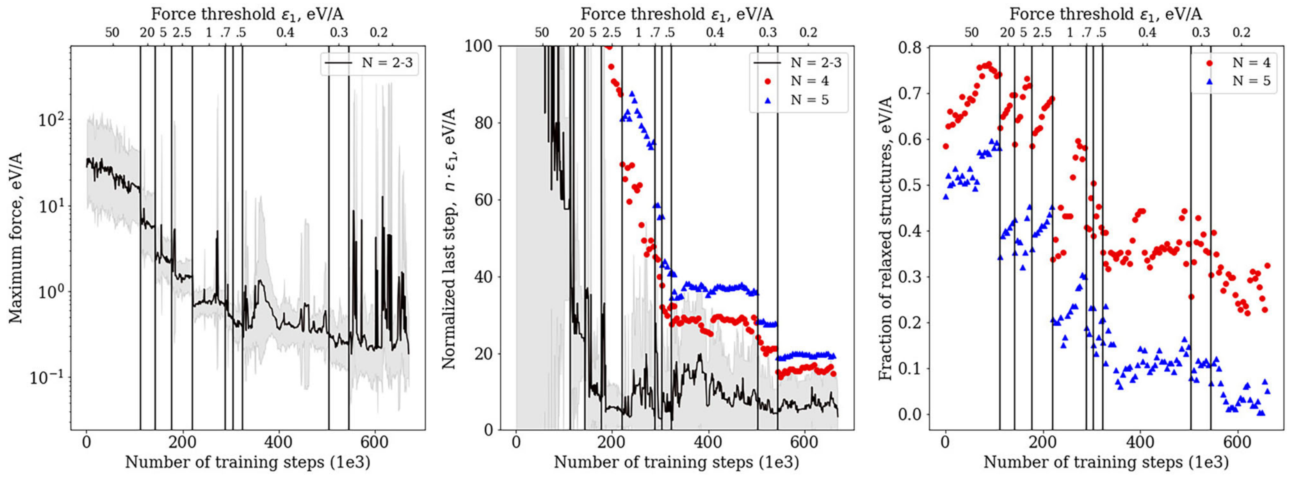

Fig. 6 | Learning curve of the TD3 model, trained on random Al and Fe configurations with $N = 2–3$ (black line) and validated on configurations with $N = 4$ (red dots) and $N = 5$ (blue triangles). We implemented a sequential decrease of the ϵ_1 parameter along with a *hybrid* reward function. The corresponding ϵ_1 values are shown on the upper axis. In every testing episode, the model was validated on datasets containing structures with $N = 4$ and $N = 5$ atoms per unit cell. For each structure, we performed 5 relaxation episodes, each starting with different initial distortions from the minimum. There were 50 structures for each N (refer to Supplementary Section 9). In the left plot, the maximum force at the last step, averaged over 20 relaxation episodes (1 per structure in the training dataset), is presented. The shaded area represents the standard deviation. The middle plot shows the normalized last step value $n \cdot \epsilon_1$ for both training and validation datasets, averaged over 20 relaxation episodes for the training dataset (1 per structure) and 250 episodes for each validation dataset (5 episodes per structure). The right plot illustrates the fraction of successful relaxation episodes out of the total number of relaxation episodes.

Agent will be capable of selecting the corresponding subspace for each atom and predicting its atomic shift based on the experience gained from relaxing smaller structures.

Finally, we assess the cumulative generalizability of the model and the applicability of algorithms for more practical tasks. Specifically, we focus on training the model on simpler structures and subsequently use it to relax more complex structures, which are larger and have new types of chemical bonds.

We use three datasets of Al-Fe structures with varying compositions generated using USPEX software. The first dataset consists of 20 structures, each containing 2–3 atoms per unit cell, and it is used to train the model. During the training process, we apply a sequential decrease of the ϵ_1 parameter, as described above, along with a *hybrid* reward function. The other two datasets, each comprising 50 structures with 4 and 5 atoms per unit cell, are used to validate the model's generalizability. In each testing episode, we relax these structures without incorporating them into the training dataset. The corresponding training curve is presented in Fig. 6 and the number of relaxation steps at the end of training in Table 1.

To evaluate changes in the quality of the model's performance for different values of ϵ_1, we use a concept of normalized last step, where the number of relaxation steps is multiplied by the force threshold to compensate for the increased difficulty as epsilon decreases. It allows to show the overall improvement of the relaxation algorithm and to compare the values between different ϵ_1-regions.

In the middle plot of Fig. 6, we observe that, on average, the normalized last step slightly changes after $\epsilon_1 = 0.3$ eV/A. This could be related to instability in the RL model, which is also reflected in the maximum force at the last step, as shown in the left plot of Fig. 6. The maximum force gradually decreases with the reduction of ϵ_1, approaching ϵ_1 as training progresses until it reaches $\epsilon_1 = 0.2$ eV/A. At this point, sharp peaks are observed, indicating possible instability due to a low sensitivity near the minimum discussed earlier. Because of this issue, switching to the *step* reward function, as discussed above, did not improve the model, and thus, we do not provide the results for this reward function here.

Regarding the model's generalizability, there is a strong correlation between the results for $N = 2–3$ and $N = 4–5$ as shown in Fig. 6. This indicates that our model can be applied to relax more complex structures without pre-training on them.

Table 1 | Number of relaxation steps up to forces 0.2 eV/A in the end of model training in Fig. 6

	***N* = 2–3**	***N* = 4**	***N* = 5**
Relaxation steps	17.3 ± 5.8	73.1 ± 39.5	97.3 ± 12.8

In the table, mean values with standard deviation are presented. For $N = 2 - 3$ the sample consisted of 20 relaxation episodes (1 per structure), for $N = 4$ and $N = 5$ it consisted of 250 episodes for each validation dataset (5 episodes per structure).

To further investigate this, we compare the structural diversity of the test and training datasets using two criteria: coordination number (CN) of atoms in the structures and the presence of structural prototypes from the training dataset for structures in the test datasets. Detailed results are provided in Supplementary Section 9. Our analysis reveals significant differences between the test and training datasets. Many coordination environments present in the test data were absent during training, and a substantial portion of test structures lack matching prototypes in the training dataset. These findings demonstrate that our model generalizes well to structurally diverse configurations.

As seen, the fraction of relaxed structures for reasonable values of ϵ_1 (~0.1 eV/A) ranges from 0.1 to 0.4, which is not a sufficient outcome. There are two primary reasons for this. The first reason is that the model needs to be trained on larger structures to efficiently identify the subspace of the most important neighbors for a given atom. The second reason is that not all interaction and symmetry patterns in structures with 4–5 atoms per unit cell are present in structures with $N = 2–3$ atoms per unit cell. A possible solution would be to incorporate an active learning approach, sequentially adding structures with higher N until the model is sufficiently trained to relax large structures. However, this approach is outside the scope of this study due to computational costs and may be considered in future research. Finally, we observed that despite additional greedy exploration (see Methods Section), the model tends to get stuck and oscillates between the same configurations near the local minimum in some test structures, failing to shift them further towards a minimum. This may be caused by the sensitivity issues discussed earlier and suggests the need for further technical development of the architecture. Specifically, the model should account for varying scales of characteristic differences (node and edge features) to accurately predict both large atomic shifts for states far from minima and small shifts close to the minimum. This issue will be addressed in future work.

Discussion

It has been demonstrated that the TD3 model, when trained on a single structure, performs comparably or even better than classical algorithms such as BFGS and CG, especially in the range of force convergence up to approximately $\epsilon_1 \sim 10^{-1} - 10^{-2}$ eV/Å. However, achieving convergence to lower force values or training on several structures simultaneously remains challenging due to the sensitivity issues of the TFN model near minima. Despite this, relaxation to forces below $10^{-1} - 10^{-2}$ eV/Å is not critically problematic in practice. A hybrid approach can be employed: after reaching this force range, one can switch to classical algorithms, which can quickly reach a minimum since the potential energy surface (PES) in this region is close to a parabolic function. Also as the results show, training the model requires approximately 100,000-1,000,000 steps, depending on the dataset size. This can be computationally expensive, particularly when using DFT for energy and force calculations. Nevertheless, we believe that training the optimizer initially on empirical potentials (e.g., EAM) or machine-learning interatomic potentials could facilitate subsequent DFT-based optimization.

The presented method can be used as a universal optimizer - once trained on a large number of structures and optimization paths, it can be applied to sufficiently similar structures without further training. This offers significant practical benefits even without strong model generalizability, as many structures are typically optimized from scratch without leveraging prior knowledge. Such a model could provide "compressed knowledge" of common optimization pathways. The algorithm might serve as a baseline optimizer for commonly used systems or for relaxing identical structures from different unstable initial configurations, e.g., in catalytic studies. Nevertheless, potential energy surfaces of molecular adsorption typically exhibit subtle differences and multiple local minima, so model's ability to identify physically meaningful minima in such scenarios requires further study. Also, our results demonstrate the efficacy of the method for Al and Fe, suggesting potential applicability to other metallic systems. However, further investigation is required to validate the algorithm's applicability across a broader range of elements.

We also explored the model's generalization capabilities from simple to complex structures. We assessed the model's performance as the number of atoms per unit cell increased. Results indicated that the model effectively relaxes supercells of similar structures, with performance improving as the size of the input structure in the training dataset increases. We anticipate that the model will eventually be capable of relaxing large structures without prior training on them, as the Agent will identify relevant subspaces for each atom and predict their atomic shifts based on experience gained from smaller structures. In addition, we found that the model can extrapolate its experience to cases where the complexity of chemical bonds increases.

These findings illustrate the potential for crystal structure relaxation using deep reinforcement learning. Overcoming the mentioned challenges could lead to a universal structure optimizer that is faster for large configurations than traditional optimizers. Nonetheless, achieving this goal requires further development and additional computational resources for model training.

Our work underscores several critical aspects of reinforcement learning. Firstly, we highlight the importance of symmetry preservation in the action space for effective model training. We show that TFN model is more suitable to structure relaxation tasks compared to CGCNN, as it limits predicted actions to spaces with the same or higher symmetry than the input structure. Secondly, we point out the limitations of training with a piecewise-constant action-value function using a *step* reward function. Direct optimization of the number of steps with this function is challenging, as the model may converge to a suboptimal policy due to a feedback loop of Critic and Actor convergence caused by the piecewise-constant nature of the true *Q*-function. However, pretraining the agent on smoother reward functions, such as *force* or *hybrid*, can mitigate this issue by increasing the likelihood of reaching terminal states early in training, a crucial condition for converging to an optimal policy.

Our study extensively examines the space of model hyperparameters and their impact on performance with detailed results presented in Supplementary Information. First, we determined that simultaneous tuning of the discount factor and noise level enables the Agent to explore efficiently while balancing short- and long-term rewards to develop an optimal policy. Higher noise increases value estimate variance, requiring a higher discount factor to prioritize long-term rewards and stabilize convergence. Conversely, low-noise settings benefit from a lower discount factor, as rewards stem primarily from the policy's immediate actions. Second, we show that the choice of exploration strategy significantly influences model performance. For equivariant models, adding random noise into the state during exploration—rather than the action—yields superior results because this approach follows symmetry restrictions of the environment. Introducing additional exploration with high noise, which we labeled as "greedy", allows to mitigate Agent stagnation in local optima, effectively preventing oscillations around suboptimal state-action regions. Finally, we investigated the influence of different reward functions. While the "*log force*" reward function enhances sensitivity near energy minima, according to our observation, it does not substantially improve performance in terms of relaxation steps for large systems. As evidenced by our tests on Al structures, presented in Supplementary Information, the "*hybrid*" reward function with weights $w_1 = 1$, $w_2 = 0$, $w_3 = 0.5$ outperforms the standard "*force*" reward function. Furthermore, as mentioned above, switching to the "*step*" reward function further minimizes the number of relaxation steps.

We compared various RL algorithms and found that TD3 is more efficient than SAC for structure relaxation tasks. However, this efficiency might be linked to exploration limitations in our SAC implementation. Future research will continue comparing different architectures and implementations of SAC and TD3 policies.

We also would like to highlight that the presented approach can be straightforwardly extended to variable-cell relaxation. To achieve this, three vectors representing changes in the cell vectors can be incorporated into the action a_t. This is a natural continuation of the current work and will be explored in future studies.

We hope our findings lay a foundation for further advancements in using reinforcement learning for structure relaxation tasks.

Methods

Preliminary of reinforcement learning

Reinforcement learning (RL) is a field of machine learning applied to decision-making problems, where an *Agent* learns to behave in a specific *Environment* by performing actions and observing the outcomes. It is most commonly formulated in the form of Markov Decision Processes (MDPs). In this framework, the Agent, at a given time t in state s_t, selects an action a_t. This action transitions the system to the next state s_{t+1}, with this transition determined by the internal dynamics of the Environment, denoted as $f(s'|s, a)$, which is also called *transition law*, so $s_{t+1} \sim f(\cdot|s_t, a_t)$. The next state s_{t+1} depends solely on the current state and action, thus satisfying the Markov property. Upon transitioning, the Agent receives a reward $r_t = R(s_t, s_{t+1}, a_t)$[31,42,43].

The decision regarding which action to take is governed by the Agent's policy, denoted here as π. The main goal of RL is to find such a policy π^*, which is called *optimal*, that maximizes an objective function $J(\pi) = \mathbb{E}_{\tau\sim\pi,f}[R(\tau)] = \mathbb{E}_{\tau\sim\pi,f}[\sum_{t=0}^{T} \gamma^t r_t]$, where $\gamma \in [0, 1]$ is the discount factor and $\tau = (s_0, a_0, s_1, a_1, \ldots)$ is the MDP trajectory, which ends at the terminal step T. The initial state s_0 is selected randomly according to a starting distribution $\rho_0(\cdot)$. The expected value is taken over the probability distribution $P(\tau|\pi) = \rho_0(s_0)\prod_{t=0}^{T-1} f(s_{t+1}|s_t, a_t)\pi(a_t|s_t)$.

In the context of algorithms used in this work the Agent maximizes not the objective $J(\pi)$, but the action-value function $Q^{\pi^*}(s, a)$. In the case of Twin

Delayed DDPG (TD3) the policy π is deterministic and $Q^{\pi^*}(s,a)$ is given by the formula:

$$Q^{\pi^*}(s,a) = \mathbb{E}_{s' \sim P(\tau|\pi^*)}\left[R(s,s',a) + \gamma \max_{a'}\left[Q^{\pi^*}(s',a')\right]\right] \tag{1}$$

In TD3 two classes of deep learning models are introduced. The first one, called *Actor*, denoted as $\pi_\theta(s)$ with tunable parameters θ, is used to approximate the optimal policy π^*. Another one, called *Critic* and denoted as $Q_\phi(s, a)$ with tunable parameters ϕ approximates the optimal action-value function $Q^{\pi^*}(s,a)$. One can find more details regarding corresponding loss functions in Supplementary Section 2 and Section 7.

Crystal structure relaxation as a Markov decision process

The structure relaxation operates within a framework of Markov decision process MDP(S,A,R), where we define each term in what follows:

- S is the state space: each $s_t \in S$ is a crystal graph[19] of the corresponding structure at time step t. In this graph, each node i represents an atom in the given unit cell and contains a node feature vector v_i. The k-th edge between two nodes $(i, j)_k$ corresponds to the bond between the i-th atom and the k-th image of its j-th neighbor and contains an edge feature vector $u_{(i,j)_k}$.
- A is the action space: each $a_t \in A$ is a graph in which each node i corresponds to the node i in s_t and contains the vector $\Delta\vec{r}_i$, representing the atomic shift: $a_t = \{\Delta\vec{r}_1, \Delta\vec{r}_2, \ldots, \Delta\vec{r}_N\}$, N is the number of atoms in a unit cell of the structure.
- R is the reward function. Four options are considered in the given work. The first one, labeled "*force*", optimizes atomic forces, which is a natural choice as it allows direct navigation to the energy minimum:

$$R_1(s_t, s_t', a_t) = -\max_{n\in[1,N]} |\vec{f}_n(s_t')|, \tag{2}$$

where $\vec{f}_n$ is the force acting on the n-th atom. The second option, labeled "*step*", optimizes the number of steps directly, which aligns with the main objective of this work:

$$R_2(s_t, s_t', a_t) = \begin{cases} -1, & \text{if } d_t = \text{False} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

where d_t is a "*done*" flag, which is "True" when the step t is terminal and "False" otherwise. The third option, called "*log force*", is more sensitive to the change of atomic forces in close vicinity to the local minimum:

$$R_3(s_t, s_t', a_t) = -\log_{10}\left(\max_{n\in[1,N]} |\vec{f}_n(s_t')|\right) \tag{4}$$

The last one, labeled "*hybrid*", is a composition of all functions above with some weights $\{w_1, w_2, w_3\}$, allowing prioritization of the described reward functions in varying proportions:

$$R_4(s_t, s_t', a_t) = \sum_{i=1}^{3} w_i R_i(s_t, s_t', a_t) \tag{5}$$

In this work, we do not employ an energy-based reward function (which is used, for example, in[18]), as this would bias the Agent toward searching specifically for the global minimum rather than any local one. This approach differs from our study's goal because crystal structures typically have relatively few energy minima, most of which are reasonably stable. We are interested in finding all of them rather than focusing solely on the global minimum.

- In this work, the "*done*" criterion for d_t aligns with those commonly used in the classical algorithms:

$$d_t = \max_{n\in[1,N]} |\vec{f}_n(s_t')| \le \epsilon_1, \tag{6}$$

where ϵ_1 is the force threshold, provided by the user.

Agent model implementation

The models for Actor and Critic were implemented using two types of networks: 1) Crystal Graph Convolutional Neural Networks (CGCNN)[19] and 2) E(3)-Equivariant Tensor-Field Neural Networks (TFN) for point clouds[20,25]. Both architectures handle graph-structured data using convolutional operations to aggregate information from neighboring nodes, capturing the correlation between target values and local interactions.

The main difference between them is that in CGCNN both node and edge feature vectors are treated as arrays of numbers, without distinguishing them by geometric characteristics. Consequently, it can be translation-invariant if node and edge features do not depend on absolute atomic positions and invariant to the O(3) group only if all features are scalars; otherwise, invariance and equivariance are not guaranteed.

In contrast, TFN is designed to maintain equivariance with respect to the whole E(3) group. All data in this model are treated as geometrical tensors, which are decomposed into irreducible representations of O(3). The convolutional layer is implemented as the direct product of irreducible representations, which guarantees equivariance of the architecture[44]. Additional information about the neural networks and graph construction can be found in Supplementary Section 1.

Implementation of exploration

Exploration in RL is the strategy by which an Agent discovers new knowledge about its Environment, choosing actions that may not yield immediate rewards but improve future decision-making. The challenge is balancing exploration with exploitation, which involves making decisions based on the current policy.

In our work, we implement exploration in TD3 using the *adding noise to state* approach to preserve the symmetry of the system (a detailed discussion of the importance of this aspect in the context of RL-based structure relaxation is provided in Supplementary Section 5). For this purpose, vectors of forces in each node are rotated around randomly generated axes $\vec{n}$ to a random angle ϕ, and the lengths of forces are modified to small values: $\vec{f}_i \to (1+\xi)\cdot(O(\vec{n},\phi)\vec{f}_i)$, where $\xi \sim U(-\lambda, \lambda)$. Here λ denotes the hyperparameter called *noise level*. It is important to note that forces in each node are altered with the same rotation and lengthening/shortening. Two settings for the noise parameter λ are utilized. In the first case λ is constant ($\lambda = c$), while in the second case, λ varies during the training episode according to the formula $\lambda(t) = (c_2 - c_1)(t/L) + c_1$, where L is the maximum number of steps per episode ($\lambda \in [c_1, c_2]$).

During training it was observed that the TD3 algorithm with exploration, described above, tends to get stuck in states with low forces, failing to shift structures further towards a minimum (see Supplementary Fig. S3, black trajectory). The Agent converges to a policy that predicts infinitesimally small actions for such states and relaxation oscillates between the same configurations. To address this issue, we proposed to include *additional greedy exploration*.

In this approach, greedy actions are introduced by adding noise to a state with a very high λ, to facilitate relevant exploration for the Agent. Greedy actions are sampled if $\max_{n\in[1,N]} |\vec{f}_n(s_t')| > f_{max}$ and $\max_{n\in[1,N]} |\vec{\Delta r}_n| < \Delta r_{max}$ for N_{gr} steps during training, where N_{gr}, Δr_{max}, and f_{max} are tunable parameters. A more detailed comparison of Agents with and without greedy exploration is provided in Supplementary Section 6.

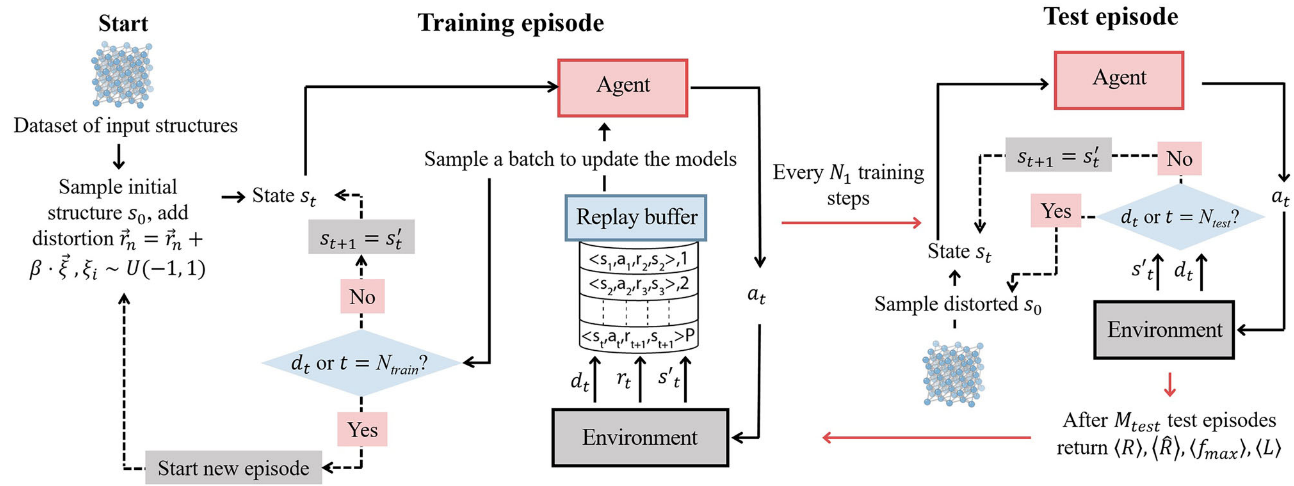
Start
Dataset of input structures
Sample initial structure s_0, add distortion $\vec{r}_n = \vec{r}_n + \beta \cdot \vec{\xi}, \xi_i \sim U(-1,1)$
State s_t
$s_{t+1} = s'_t$
No
d_t or $t = N_{train}$?
Yes
Start new episode
Training episode
Agent
Sample a batch to update the models
Replay buffer
$<s_1,a_1,r_2,s_2>,1$
$<s_2,a_2,r_3,s_3>,2$
$<s_t,a_t,r_{t+1},s_{t+1}>P$
d_t r_t s'_t
Environment
a_t
Every N_1 training steps
Test episode
Agent
$s_{t+1} = s'_t$
No
Yes
d_t or $t = N_{test}$?
State s_t
Sample distorted s_0
s'_t d_t
Environment
a_t
After M_{test} test episodes return $\langle R\rangle, \langle\hat{R}\rangle, \langle f_{max}\rangle, \langle L\rangle$

Fig. 7 | Workflow of the reinforcement learning algorithm used for structure relaxation.

Algorithm workflow

The class for the Environment is implemented according to the standards of *Gym*[45] Python library and constructed with Python Materials Genomics (*pymatgen*)[46] and Atomic Simulation Environment (*ASE*)[47] libraries.

The training loop of the RL algorithm used for structure relaxation is illustrated in Fig. 7. Before initiating the training of the Agent, the structures in the provided dataset undergo relaxation with a classical algorithm, such as BFGS.

The Agent undergoes training across M_{train} episodes, each of which lasts a maximum of N_{train} steps. At the beginning of each episode, a crystal structure is selected from the data set and disturbed from equilibrium by introducing random distortions into the atomic coordinates: $\vec{r}_n(s_0) = \vec{r}_n^{\,*} + \beta \cdot \vec{\xi}_n$, where $(\vec{\xi}_n)_i \sim U(-1,1)$, $\vec{r}_n^{\,*}$ are the coordinates of the n-th atom in equilibrium. Here, β is referred to as the *distortion parameter*. This distorted state serves as the initial point in the trajectory. Then, at each step the Agent takes actions, shifts atoms in the structure, gets the next state s'_t and "*done*" flag d_t, stores this transition in the storage called *Replay Buffer,* and, if there are enough transitions in the Replay Buffer, updates Actor and Critic models. The episode ends when "*done*" criterion is met or when the time limit N_{train} is reached.

Every N_1 time steps training is paused to conduct M_{test} test episodes, each lasting maximum N_{test} steps. The testing episodes are similar to the training ones, the only difference is that the models are not updated and the transitions are not stored in the Replay Buffer. During testing, the following metrics are measured and averaged over all episodes:

- Cumulative discounted score $R(\tau) = \sum_{t=0}^{T} \gamma^t r_t$;
- Cumulative full score $\hat{R}(\tau) = \sum_{t=0}^{T} r_t$;
- Maximum force at the last step $\max_{n\in[1,N]} |\vec{f}_n(s_T)|$ (further referred as *maximum force*);
- Number of relaxation steps (further referred as *relaxation steps*).

The corresponding averages over M_{test} episodes are denoted as $\langle R\rangle$, $\langle\hat{R}\rangle$, $\langle f_{max}\rangle$, $\langle L\rangle$.

In this work, $N_1 = 1000$, $N_{\text{test}} = 100$, $N_{\text{train}} = 1000$, but these parameters can be changed by the user.

Data availability

Cif files for structures generated with USPEX can be found and downloaded at: https://github.com/ElenaTrukhan/RL_structure_relaxation.

Code availability

The code that was used in the findings of this study is available from https://github.com/ElenaTrukhan/RL_structure_relaxation.

Received: 28 December 2024; Accepted: 3 July 2025;
Published online: 24 September 2025

References

1. Pulay, P. Convergence acceleration of iterative sequences. the case of SCF iteration. *Chem. Phys. Lett.* **73**, 393–398 (1980).
2. Press, W. H., Flannery, B., Teukolsky, S. & Vetterling, W. *Numerical recipices, the art of scientific computing* (Cambridge U. Press, Cambridge, MA, 1986).
3. Banadaki, A. D., Tschopp, M. A. & Patala, S. An efficient Monte Carlo algorithm for determining the minimum energy structures of metallic grain boundaries. *Comput. Mater. Sci.* **155**, 466–475 (2018).
4. Pillardy, J., Arnautova, Y. A., Czaplewski, C., Gibson, K. D. & Scheraga, H. A. Conformation-family Monte Carlo: a new method for crystal structure prediction. *Proc. Natl. Acad. Sci. USA* **98**, 12351–12356 (2001).
5. Parrinello, M. & Rahman, A. Crystal structure and pair potentials: a molecular-dynamics study. *Phys. Rev. Lett.* **45**, 1196–1199 (1980).
6. Podryabinkin, E. V., Tikhonov, E. V., Shapeev, A. V. & Oganov, A. R. Accelerating crystal structure prediction by machine-learning interatomic potentials with active learning. *Phys. Rev. B.* **99**, 064114 (2019).
7. Foerster, D. & Gueddida, S. A low resources space time approach to the GW approximation. *Comput. Mater. Sci.* **187**, 110078 (2021).
8. Strout, D. L. & Scuseria, G. E. A quantitative study of the scaling properties of the Hartree-Fock method. *J. Chem. Phys.* **102**, 8448–8452 (1995).
9. Lenz, M.-O. et al. Parametrically constrained geometry relaxations for high-throughput materials science. *npj Comput. Mater.* **5**, 123 (2019).
10. Pan, E., Karpovich, C. & Olivetti, E. Deep reinforcement learning for inverse inorganic materials design. *npj Comput. Mater.* **10**, 287 (2024).
11. Sui, F., Guo, R., Zhang, Z., Gu, G. X. & Lin, L. Deep reinforcement learning for digital materials design. *ACS Mater. Lett.* **3**, 1433–1439 (2021).
12. Manna, S. et al. Learning in continuous action space for developing high dimensional potential energy models. *Nat. Commun.* **13**, 368 (2022).
13. Zhou, Z., Kearnes, S., Li, L., Zare, R. N. & Riley, P. Optimization of molecules via deep reinforcement learning. *Sci. Rep.* **9**, 10752 (2019).
14. Wang, X. et al. Towards efficient discovery of green synthetic pathways with Monte Carlo tree search and reinforcement learning. *Chem. Sci.* **11**, 10959–10972 (2020).
15. Vasudevan, R. K., Orozco, E. & Kalinin, S. V. Discovering mechanisms for materials microstructure optimization via reinforcement learning of a generative model. *Mach. Learn. Sci. Technol.* **3**, 04LT03 (2022).

16. Modee, R., Mehta, S., Laghuvarapu, S. & Priyakumar, U. D. MolOpt: autonomous molecular geometry optimization using multiagent reinforcement learning. *J. Phys. Chem. B.* **127**, 10295–10303 (2023).
17. Barrett, R. & Westermayr, J. Reinforcement learning for traversing chemical structure space: optimizing transition states and minimum energy paths of molecules. *J. Phys. Chem. Lett.* **15**, 349–356 (2024).
18. Bihani, V., Manchanda, S., Sastry, S., Ranu, S. & Krishnan, N. M. A. StriderNet: A Graph Reinforcement Learning Approach to Optimize Atomic Structures on Rough Energy Landscapes. In *Proceedings of the 40th International Conference on Machine Learning*, Vol. 202, 2431–2451 (PMLR, 2023).
19. Xie, T. & Grossman, J. C. Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties. *Phys. Rev. Lett*. **120**, 145301 (2018).
20. Thomas, N. et al. Tensor field networks: Rotation- and translation-equivariant neural networks for 3d point clouds. Preprint at https://arxiv.org/abs/1802.08219 (2018).
21. Park, C. W. & Wolverton, C. Developing an improved crystal graph convolutional neural network framework for accelerated materials discovery. *Phys. Rev. Mater.* **4**, 063801 (2020).
22. Laugier, L. et al. Predicting thermoelectric properties from crystal graphs and material descriptors - first application for functional materials. Preprint at https://arxiv.org/abs/1811.06219 (2018).
23. Deng, B. et al. CHGNet as a pretrained universal neural network potential for charge-informed atomistic modelling. *Nat. Mach. Intell.* **5**, 1031–1041 (2023).
24. Gong, S. et al. Examining graph neural networks for crystal structures: limitations and opportunities for capturing periodicity. *Sci. Adv.* **9**, eadi3245 (2023).
25. Batzner, S. et al. E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nat. Commun*. **13**, 2453 (2022).
26. Batatia, I., Kovacs, D. P., Simm, G. N. C., Ortner, C. & Csanyi, G. MACE: higher order equivariant message passing neural networks for fast and accurate force fields. *Adv. Neural Inf. Process. Syst.* **35**, 11423–11436 (2022).
27. Smidt, T. E., Geiger, M. & Miller, B. K. Finding symmetry breaking order parameters with Euclidean neural networks. *Phys. Rev. Res.* **3** L012002 (2021).
28. Chen, Z. et al. Direct prediction of phonon density of states with Euclidean neural networks. *Adv. Sci.* **8**, e2004214 (2021).
29. Koker, T., Quigley, K., Taw, E., Tibbetts, K. & Li, L. Higher-order equivariant neural networks for charge density prediction in materials. *Npj Comput. Mater*. **10**, 161 (2024).
30. Fujimoto, S., van Hoof, H., Meger, D. Addressing Function Approximation Error in Actor-Critic Methods. In *Proceedings of the 35th International Conference on Machine Learning*, Vol. 80, 1587–1596 (PMLR, 2018).
31. Winder, P.*Reinforcement Learning: Industrial Applications of Intelligent Agents* (O'Reilly Media, Inc., 2021).
32. Haarnoja, T., Zhou, A., Abbeel, P. & Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, Vol. 80, 1861–1870 (PMLR, 2018).
33. Ackland, G. J., Bacon, D. J., Calder, A. F. & Harry, T. Computer simulation of point defect properties in dilute Fe-Cu alloy using a many-body interatomic potential. *Philos. Mag. A.* **75**, 713–732 (1997).
34. Mendelev, M., Srolovitz, D., Ackland, G. & Han, S. Effect of Fe segregation on the migration of a non-symmetric $\Sigma5$ tilt grain boundary in Al. *J. Mater. Res.* **20**, 208–218 (2005).
35. Zope, R. R. & Mishin, Y. Interatomic potentials for atomistic simulations of the Ti-Al system. *Phys. Rev. B.* **68**, 024102 (2003).
36. Broyden, C. G. The convergence of a class of double-rank minimization algorithms 1. General considerations. *IMA J. Appl. Math.* **6**, 76–90 (1970).
37. Fletcher, R. A new approach to variable metric algorithms. *Comput. J.* **13**, 317–322 (1970).
38. Goldfarb, D. A family of variable-metric methods derived by variational means. *Math. Comput.* **24**, 23–23 (1970).
39. Shanno, D. F. Conditioning of quasi-Newton methods for function minimization. *Math. Comput.* **24**, 647–656 (1970).
40. Hestenes, M. R. & Stiefel, E. Methods of conjugate gradients for solving linear systems. *J. Res. Natl Bur. Stand.* **49**, 409 (1952).
41. Glass, C. W., Oganov, A. R. & Hansen, N. USPEX-evolutionary crystal structure prediction. *Comput. Phys. Commun.* **175**, 713–720 (2006).
42. Sutton, R. S. & Barto, A. G. *Reinforcement Learning: An Introduction* (Cambridge U. Press, 1986).
43. Achiam, J. Spinning up in deep reinforcement learning. https://spinningup.openai.com/en/latest/ (2018).
44. Geiger, M. et al. Euclidean neural networks: e3nn (2022).
45. Brockman, G. et al. OpenAI Gym. Preprint at https://arxiv.org/abs/1606.01540 (2016).
46. Ong, S. P. et al. Python Materials Genomics (pymatgen): a robust, open-source Python library for materials analysis. *Comput. Mater. Sci.* **68**, 314–319 (2013).
47. Larsen, A. H. et al. The atomic simulation environment-A Python library for working with atoms. *J. Phys. Condens. Matter* **29**, 273002 (2017).

Acknowledgements
This work was supported by the Russian Science Foundation (grant #19-72-30043). The calculations were performed on the Zhores cluster at the Skolkovo Institute of Science and Technology.

Author contributions
E.T. implemented the code, trained the reinforcement learning (RL) models, analyzed the experimental results, and wrote the manuscript; E.M. contributed to the code implementation; A.O. and E.M. supervised the work; all authors discussed the results and reviewed the manuscript.

Competing interests
The authors declare no competing interests.

Additional information
Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41524-025-01731-1.

Correspondence and requests for materials should be addressed to Elena Trukhan.