Automated Construction of Neural Network Potential Energy Surface: The Enhanced Self-Organizing Incremental Neural Network Deep Potential Method

Mingyuan Xu[1], Tong Zhu[1,2]* and John Z. H. Zhang[1,2,3,4]*

[1]*Shanghai Engineering Research Center of Molecular Therapeutics & New Drug Development, Shanghai Key Laboratory of Green Chemistry & Chemical Process, School of Chemistry and Molecular Engineering, East China Normal University, Shanghai 200062, China*

[2]*NYU-ECNU Center for Computational Chemistry at NYU Shanghai, Shanghai 200062, China*

[3]*Department of Chemistry, New York University, NY, NY 10003, USA*

[4]*Collaborative Innovation Center of Extreme Optics, Shanxi University, Taiyuan, Shanxi 030006, China*

*Correspondence should be addressed to: tzhu@lps.ecnu.edu.cn or john.zhang@nyu.edu

Abstract

In recent years, the use of deep learning (neural network) potential energy surface (NNPES) in molecular dynamics simulation has experienced explosive growth as it can be as accurate as quantum chemistry methods while being as efficient as classical mechanic methods. However, the development of NNPES is highly non-trivial. In particular, it has been troubling to construct a dataset that is as small as possible yet can cover the target chemical space. In this work, an ESOINN-DP method is developed, which has the enhanced self-organizing incremental neural-network (ESOINN) and a newly proposed error indicator at its core. With ESOINN-DP, One can construct the NNPES with little human intervention, and this method ensures that the constructed reference dataset covers the target chemical space with minimum redundancy. The performance of the ESOINN-DP method has been well validated by developing neural network potential energy surfaces for water clusters and by de-redundancy of a sub-data set of the ANI-1 database. We believe that the ESOINN-DP method provides a novelty idea for the construction of NNPES and especially, the reference datasets, and it can be used for MD simulations of various gas-phase and condensed-phase chemical systems.

1. Introduction

Theoretical modeling has become one of the most important tools in the study of biomolecules. Relevant modeling approaches mainly fall into two categories: one based on quantum mechanical (QM) algorithms and the other based on classical molecular mechanics (MM) or empirical interatomic potentials (force fields). The MM methods have the advantage in terms of computational efficiency, but their accuracy is often questioned as they lack important quantum effects such as polarization and charge transfer. The QM calculations are usually more rigorous than MM methods and can offer the capability to accurately model all chemistries and chemical environments in principle. However, its applications are significantly limited by its computational cost. To some extent, efficiency and accuracy of theoretical methods seem to be contradictory.

Recently, the development of machine learning potential energy surfaces (ML PES) has made it possible to solve this contradiction. In its early stage, ML PESs were only applicable to small systems with several atoms[1-6]. Since Behler and Parrinello proposed the high dimensional neural network potentials[7, 8], a series of ML approaches which are suitable for large systems that contain thousands or even millions of atoms have been proposed.[9] For example, the Gaussian Approximation Potentials of Csányi and co-workers[10], the GDML and DTNN methods of Müller et al.[11, 12], the kCON model of Hammer et al.[13], and the Deep Potential (DP) model of E and coworkers[14]. There are several machine learning methods that can be used to build the molecular PES, including the kernel ridge regression (KRR), support vector regression (SVR), and multilayer neural networks (NN)[15]. Among them, the artificial NN is the most promising one, which in principle is able to approximate any real-valued function to arbitrary accuracy[16]. And it has been used in many important physical or chemical questions[17-29].

Although neural-network (or deep-learning) based PES (NNPES or deep potential, DP) has achieved great success, there are still several key issues that need to be carefully considered in its applications.[30, 31] (1), The construction of reference dataset that covers the target chemical space while being as small as possible. (2), The selection proper molecular descriptors which can represent the chemical environment while with minimal complexity. (3), The selection of hyper-parameters for NN. (4), the reliability assessment of trained model. A series of outstanding works targeting the solution of these questions have been reported recently[32-38]. For the construction of reference dataset, Collins suggested to employ MD simulation to sample the molecular configurations[39]. Raff and coworkers proposed a probabilistic scheme to construct reference dataset by selecting new configurations based on the present density of points available in the current region of configuration space[40]. In their previous work, Artrith et al. suggested using multiple NN models to identify poorly sampled regions of the configuration space.[41] In this scheme, a set of different NN PES models will be trained for a given training data set. And a large number of trial configurations are evaluated by these models. If a given

structure differs obviously from all trained data, the predictions of these models should be significantly different. Conversely, if the training set already has similar structures of the given one, the predicted results of these models should be consistent. This algorithm was also called "active learning" and has been used by many works[42-45]. Recently, a concurrent learning algorithm was proposed by E and co-workers[46]. Compared with active learning, this method can construct the reference datasets on the fly without any pre-prepared data.

There are also several useful discussions about the selection of molecular descriptors.[7, 15, 47, 48] The selection of hyper-parameters is highly non-trivial, since it can only be done through trial-and-error or blind guessing. And it has become a major obstacle in the application of high-dimensional NN PESs. Recently, Behler and coworkers discussed the automatic selection of hyper-parameters in molecular descriptors, which solves these two issues to some extent.[48, 49] However, there are still many hyper-parameters such as learning rate and network structure parameters which have huge impact on the accuracy and efficiency of neural networks. The choice of these parameters is also largely empirically dominated.

In this work, to address the above-mentioned four issues, we proposed an automated NNPES training framework named enhanced self-organizing incremental neural-network deep potential (ESOINN-DP). This method has three important features: (1), the automated construction of the reference dataset that requires little human intervention and with low redundancy. (2), the automated optimization of neural network structures. (3), the self-verification of trained models. The ESOINN-DP method has proven to be effective in simplifying the training process of NNPES and saving computational resources. This paper is organizing as follows. In sections 2, we present the basic methodology of algorithms employed by ESOINN-DP. Then we evaluate its performance by several examples in section 3. Finally, conclusions and outlooks are given in the last section.

2. Theory and method

To achieve the automated construction of the reference dataset, the ESOINN-DP method employs an enhanced self-organizing incremental neural-network (ESOINN) layer to enable the on-the-fly collection of data. Inspired by the structure of the cerebral cortex, the ESOINN was proposed to accomplish online unsupervised learning tasks. In the cerebral cortex, different areas are responsible for different stimuli, such as sight, hearing, and smell. In ESOINN-DP, ESOINN is responsible for sensing the input patterns (For simplicity, we use "pattern" denotes molecules and their configurations.) and classifying them based on structural similarity. These patterns will then be passed to the neural networks which are called meta-NNs in the DP layer according to the similarity between them and nodes in the ESOINN layer. For each pattern, its potential energy, atomic forces, atomic charges, and dipole will be predicted by user-defined number of different meta-NNs (usually 3~4). A decision layer will collect the predictions of

these different meta-NNs, analyze their consistency, and give the final results (include the final prediction and an error-indicator which reflects its reliability).

The architecture of the ESOINN-DP method is shown in Figure 1. Details of these algorithms were discussed as follows.

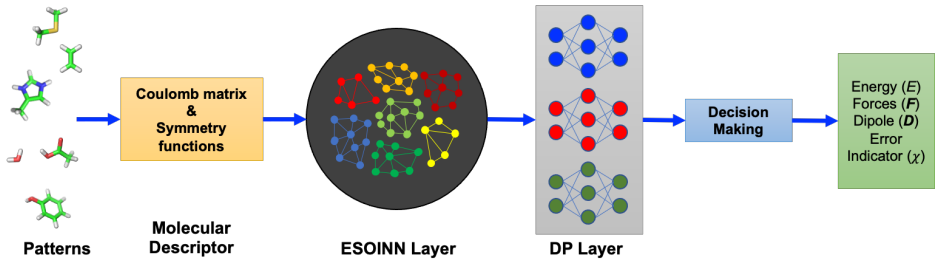

Figure 1. The architecture of the ESOINN-DP model.

2.1 Molecular descriptors

In ESOINN-DP, each pattern in the reference dataset is represented by two set of descriptors: the sorted eigenvalue spectrum of the Coulomb matrix[50] and symmetry functions[51]. The employment of Coulomb matrix can easily convert 3D structures to 1D vectors while keeping the translation, rotation, and permutational symmetry.

To make eigenvalues of all molecules equal in length, virtual atoms are introduced. The definition of Coulomb matrix is

$$C_{ij} = \begin{cases} 0.5Z_i^{2.4} & \forall i = j \\ \frac{Z_i Z_j}{|R_i - R_j|} & \forall i \neq j \; and \; i \notin virtual \; atoms \\ 0 & \forall i \neq j \; and \; i \in virtual \; atoms \end{cases} \quad (1)$$

In addition to the Coulomb matrix, the symmetry functions (*S*) developed by Isayev and co-workers were used as the molecular descriptor in the DP layer. *S* consists the radial and angular parts as shown in Eq (2) and Eq (3).

$$S(radial) = \sum_{j \neq i} e^{-\eta(R_{ij} - R_s)^2} f_c(R_{ij}) \quad (2)$$

$$S(angular) = 2^{1-\zeta} \sum_{j \neq i, j \neq k} \left(1 + cos(\theta_{ijk} - \theta_s)\right)^{\zeta} \times e^{-\eta\left(\frac{R_{ij} + R_{ik}}{2} - R_s\right)^2} f_c(R_{ij}) f_c(R_{ik}) \quad (3)$$

ζ, θ_s, and η are hyper-parameters that controls the diversity of symmetry functions. By adjust them one can improve the predictive power of sub-NNs in the DP layer.

It should be noted that the symmetry functions can replace the Coulomb matrix in principle. However, the relatively simple Coulomb matrix can already reflect the structural difference of molecules in the dataset, which means that its employment can improve the efficiency of the ESOINN obviously while doesn't reduce the fidelity significantly. Because the training of

molecular properties such as potential energy and atomic forces needs higher requirements on accuracy, so the more complex symmetric functions were chosen in the DP layer.

2.2 The ESOINN Layer

The ESOINN layer contains an enhanced self-organizing incremental neural network (ESOINN) which had been proposed to accomplish on-the-fly unsupervised learning tasks.[52, 53] It is suitable for learning the representation of the topology structure of high dimensional data with a manner of incremental learning which addresses the ability of repeatedly training a network using new data without destroying the learned knowledge.

Once the ESOINN layer is trained, we can get a network model that contains a set of classified nodes and the connection relationships between them. Each node represents some of the input patterns and has three important features: the weight vector W, mean accumulated points h, and edges between it and its neighboring nodes. The weight vector W_s of node s is a 1D vector of equal length to the sorted Coulomb matrix eigenvalue spectrum in the reference dataset. In fact, it can also be converted to a pattern. The mean accumulated points h_s reflect the density information of node s (the occurrences of similar structures of node s in learning process). The edges represent the similarity between different nodes. The actual training process is the addition and deletion of nodes in the network, and the adjustment of the weight vectors and edges. Through the ESOINN layer, we can easily locate the structural subset with higher redundancy and expand the subset with low redundancy in a targeted manner, and explore new subsets. The training process of the ESOINN layer is shown in Figure 2. A brief description of the training process is as following.

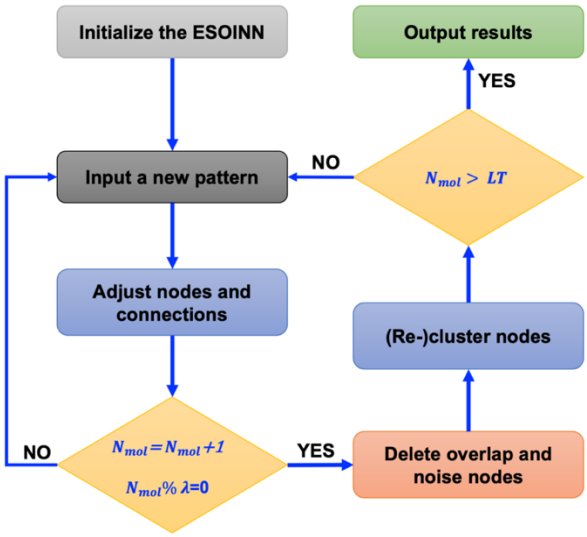

Figure 2. The learning process of the ESOINN Layer.

Here, it is assumed that the complete training process includes LT learning times (LT is the total number of patterns input into ESOINN) totally. Each time ESOINN accepts an input

pattern and completes an adjustment, it is regarded as one time of learning. Every λ (λ = 500 in this work) times of learning is regarded as a learning cycle. The learning process of ESOINN can be divided into 3 steps:

(1), initialize the ESOINN. Before the training process, the ESOINN layer needs to be initialized. In the beginning, the network contains only two connected nodes. The weight vector of each node is randomly selected from the initial dataset. While the initial dataset is constructed by the user, which may contain dozens of patterns. As the content and size of the initial dataset will not influent the final results, it can be obtained easily.

(2), input new patterns and adjust nodes and their connections (edges) accordingly. When we input a new pattern (denotes by ε) to ESOINN, we need to determine whether to insert a new node to represent it by finding its first-nearest node s_1 and second-nearest node s_2:

$$s_1 = argmin_{s\in N}\|\varepsilon - W_s\| \tag{4}$$

$$s_2 = argmin_{s\in N\setminus s_1}\|\varepsilon - W_s\| \tag{5}$$

where N is the set of all nodes in the current ESOINN layer.

A new node will be added to represent pattern ε if the distance between it and s_1 or s_2 is greater than a similarity threshold T (which will be trained in the training process). And then the connection relationship of node s_1 and s_2 will be adjusted.

The mean accumulated points (node density) of s_1 will also be updated. In ESOINN, the node density of node s can be expressed as

$$h_s = \bar{q}_s = \frac{1}{N_{win}}\sum_{j=1}^{n}\left(\sum_{i=1}^{\lambda} p_s\right) \tag{6}$$

where n represents the number of learning cycles experienced, N_{win} represents the number of wins of node s. q_s is the accumulation of points p_s in the whole learning process and p_s is related to the average distance ($\overline{d_s}$) between node s and the its surrounding nodes which can be expressed by

$$p_s = \begin{cases} 1/\left(1+\overline{d_s}\right)^2 & if\ node\ s\ is\ winner \\ 0 & if\ node\ s\ is\ not\ winner \end{cases} \tag{7}$$

Then, the weight vector of winner node s_1 and its neighboring node (j)will be updated:

$$\Delta W_{s1} = \frac{1}{N_{win}}\left(\varepsilon - W_{s_1}\right) \tag{8}$$

$$\Delta W_j = \frac{1}{100N_{win}}\left(\varepsilon - W_{s_1}\right) \tag{9}$$

(3), delete noise nodes and classify node to different classes.

In ESOINN, all nodes that can be connected by edges are considered belong to a subclass. If the number of patterns input so far is an integer multiple of λ, the subclass label of every node will be updated and noise nodes which has few connections will be deleted. If the learning

process is finished, all nodes will be classified to different subclasses and ESOINN will report the number of subclasses and their prototype vectors.

Following above algorithms, the network will gradually grow and adjust to learn new information. In each learning round, only some of the nodes' weights and the connection relations (edges) between them are updated, which ensures the efficiency. More details of these algorithms can be found in Ref. 50 and Ref. 51.

2.3 Self-verification of ESOINN-DP

In MD simulations driven by NNPES, it is crucial to judge the reliability of the potential on-the-fly to ensure the accuracy of the simulation results. In ESOINN-DP, an uncertainty evaluation algorithm was proposed to achieve this purpose. After training, we can get N disconnected subclasses and their similarity relationships in the ESOINN layer. At the DP layer, N neural networks called meta-NNs are placed. For a given structure, we first describe it with symmetry functions and get the corresponding pattern (denoted by R_t). Then, m different meta-NNs that correspond to the m closest subclasses to its sorted eigen spectrum of Coulomb matrix ε_t are trained to predict its properties (Fig. 3). In this way, each structure will be learned by multiple NNs, and the training set of each NN is different, which eliminates the possibility of consistency errors in multiple networks.

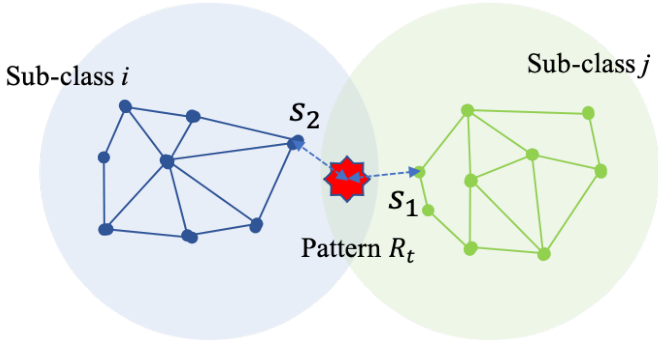

Figure 3. For a given structure R_t with eigen spectrum ε_t, 2 meta-NNs corresponding to the 2 nearest subclass i, j, k to R_t labeled by ESOINN will be trained to predict its properties.

In addition, if there are already structures in the current dataset that are very similar to R_t, then these meta-NNs should give consistent predictions. If, on the other hand, R_t is a completely new structure or one that occurs very infrequently in the current dataset, the meta-NNs should give very different predictions. Inspired from this uncertainty principle, we design an error indicator which can report the error between labeled and predicted properties quantitatively.

For a meta-NN net_i, the deviation D_{net_i} between its predictions and the labeled data should obey the gaussian distribution:

$$D_{net_i} \sim N\left(\mu_{net_i}, \sigma^2_{net_i}\right) \tag{10}$$

μ and σ are the mathematical expectation and standard deviation of the normal distribution. For well-trained models, μ should be zero.

The average prediction of the potential energy of R_t from an ensemble of meta-NNs are

$$E(R_t) = \langle E_{net_i}(R_t)\rangle_{net_i}, i = 1,2, \dots m \tag{11}$$

While the average prediction of the atomic forces of atom j in R_t are

$$\boldsymbol{F}(j, R_t) = -\nabla_j E(R_t) = \langle \boldsymbol{F}_{net_i}(j, R_t)\rangle, i = 1,2, \dots m \tag{12}$$

Due to the training processes of meta-NNs are independent to each other, the deviation between the average prediction and the reference value should also obey the gaussian distribution:

$$D_{final} \sim N\left(\langle \mu_{net_i}\rangle, \sum_i \sigma^2_{net_i}\right), i = 1,2, \dots m \tag{13}$$

In the case of force prediction, the unsigned error of the final predictions D_{Force} of R_t by an ensemble of meta-NNs is always greater or equal to 0, it should obey the folded normal distribution and can be expressed as follows:

$$|D_{Force}(j, R_t)| = \left\|\langle \boldsymbol{F}_{net_i}(j, R_t)\rangle - \boldsymbol{F}_{ref}(j, R_t)\right\| \sim 2N\left(0, \sum_{net_i} \sigma^2_{Force, net_i}\right), i = 1,2, \dots m \tag{14}$$

Based on Eq (10) and Eq (14), an error indicator χ_t is defined and used to quantitatively estimate the prediction power of meta-NNs.

$$\begin{aligned}\chi_t &= \max\left\|\boldsymbol{F}_{net_i}(j, R_t) - \langle \boldsymbol{F}_{net_i}(j, R_t)\rangle\right\| \\ &= \max\left\|\boldsymbol{F}_{net_i}(j, R_t) - \boldsymbol{F}_{ref}(j, R_t) - \left(\langle \boldsymbol{F}_{net_i}(j, R_t)\rangle - \boldsymbol{F}_{ref}(j, R_t)\right)\right\| \\ &= \max\left\|D_{Force, net_i}(j, R_t) - D_{Force}(j, R_t)\right\|\end{aligned} \tag{15}$$

The distribution of $D_{Force,net_j} - D_{Force}$ should obey the gaussian distribution of $N\left(0, \left(\sum_{net_k,} \sigma^2_{Force,net_k}\right) + \sigma^2_{Force,net_i}\right)$, which is wider than the distribution of $D_{Force}(j, R_t)$ as shown in Eq (14). In ESOINN-DP, we use χ_t as a criterion to determine whether a pattern R_t can be recognized by existing models.

$$R_t = \begin{cases} known & if\ 0 < \chi_t \le \delta_{Force} \\ questionable & if\ \delta_{Force} < \chi_t \le 2\delta_{Force} \\ unknown & if\ \chi_t > 2\delta_{Force} \end{cases} \tag{16}$$

where δ_{Force} is a user-defined value which represents the expected precision of the NNPES.

In summary, when ESOINN-DP get a new pattern, the ESOINN layer will send this pattern to meta-NNs in the DP layer according to the similarity between the input and existing reference

datasets. Then the decision layer will evaluate the predicts from the DP layer and give the final results. By combining this self-validation algorithm with ESOINN, one can locate new data, ensure that the reference dataset has very low redundancy, and then automatically construct NNPES. More importantly, one can purposefully expand the reference dataset to cover the target chemical space in the concurrent learning manner.

2.4 Optimization of hyper-parameters

In the development of NNPES, the selection and optimization of hyperparameters is very important and difficult. In ESOINN-DP, the DP layer contains many meta-NNs which are trained many times in the concurrent learning process and the reference dataset is gradually expanded at the same time. To further accelerate the training process, we employ the genetic algorithms to optimize the structural parameters (number of neurons) of meta-NNs in each round of ESOINN training.

(1) First, an initialized set of structural parameters $A_1 = [H_1^1, H_1^2, \dots H_1^N]$ is selected as the best candidate structure $A_{c,j}$ where H_1^N represents the number of neurons in the Nth hidden layer of the first round.

(2) Based on the candidate structural parameter A_c from the previous round, generate structural parameters for meta-NNs. The structure parameter of meta-network i in round j is expressed as follows:

$$A_{i,j} = A_{c,j} + \gamma_i \cdot e_{c,j} \tag{17}$$

γ is a vector of random integers between -5 and 5 with the same dimension as $A_{i,j}$. And $e_{c,j}$ is a base vector made up of 10 percent of each dimension of $A_{c,j}$.

(3) Train the meta-NNs. Each meta-NN in the ESOINN-DP model is independently trained until the RMSE of atomic forces between predictions of meta-networks and reference values is less than predefined training accuracy.

The parameter number C_i of meta-network in the j^{th} round is calculated by the following formula:

$$C_{i,j} = I \cdot H_{i,j}^1 + \sum_{n=1}^{N-1} H_{i,j}^n \cdot H_{i,j}^{n+1} + H_{i,j}^N \cdot O \tag{18}$$

Among them, I represents the input dimension of meta-network, and O represents the output dimension. Then the training of the j^{th} round selects the smallest network structural parameter as the best candidate parameter A_c of the $(j+1)^{th}$ round.

(4) The optimization is completed until the difference between each dimension of $A_{c,j}$ and $A_{c,j+1}$ is less than 10%.

We need to emphasize that since the ESOINN-DP training set may be built faster than the optimization of hyper-parameter. In these cases, it is controlled by user whether stop the hyper-parameter optimization.

2.5 Automated construction of ESOINN-DP potential models

The workflow of the ESOINN-DP is shown in Figure 4.

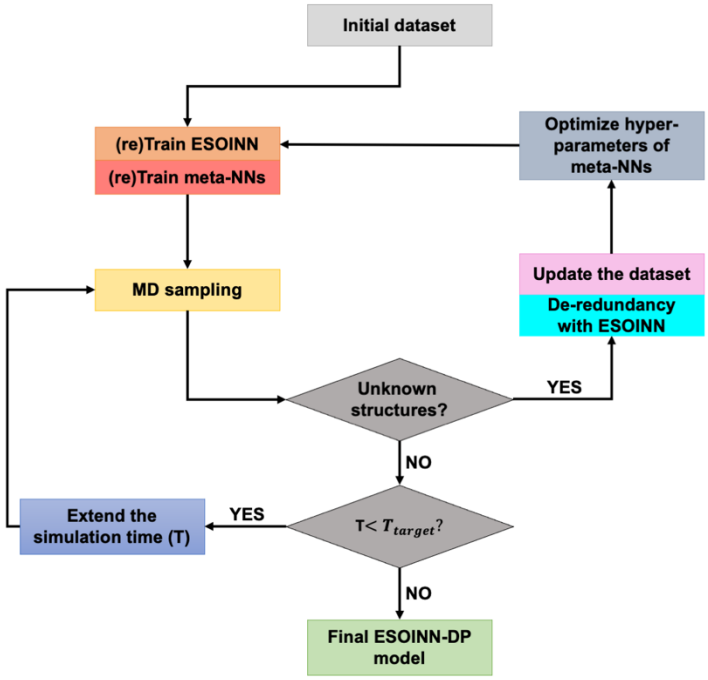

Figure 4. The workflow of auto construction of ESOINN-DP.

Detailed steps are listed as follows:

(1) Construct an initial dataset.

(2) Train an initial ESOINN-DP model. The ESOINN layer will recognize main patterns in the reference dataset and provide the density information of each node.

(3) Perform enhanced sampling with the ESOINN-DP model to expand the reference dataset.

There are two basic enhanced sampling strategies in the ESOINN-DP method. One strategy is to select multiple structures from low-density regions to perform temperature parallel MD simulations in each round of concurrent learning. Another is to select the main difference between the main patterns in the ESOINN layer as collective variables (CVs) and perform meta-dynamics simulations or target MD simulations. In this work, we use the first strategy as an example.

(4) Evaluate the performance of the trained DP models.

As discussed above, the error indicator χ_t for each pattern R_t were calculated on the fly to estimate the reliability of predictions and find the questionable and unknown patterns.

(5) If there are no unknown patterns detected from MD simulations in the current round of concurrent learning, extend the simulation time T until the target length (T_{target}) is reached. Then go to step 8. Otherwise, go to step 6.

(6) Remove the redundancy in the sampled unknown patterns.

In ESOINN-DP, there are two strategies to reduce the redundancy in the sampled unknown patterns.

First, the unknown and questionable structures encountered in the MD simulation will be collected and sent to ESOINN layer to determine which pattern should be added to the training set. In this way, ESOINN-DP avoid the redundancy in the reference dataset. Second, the ESOINN layer can focus more on inter-class patterns to increase the diversity of reference dataset. In ESOINN-DP, a new input signal ε will be labeled as "normal", "candidate" and "unimportant" according to the distance to its winner node s_1 and second winner node s_2.

1) If the distance between ε and s_1 or s_2 is greater than two times of the corresponding similarity threshold T_i, ε will be labeled as a "unimportant" signal.
2) If s_1 and s_2 are belonging to the same classification and the distance between input s_1 and s_2 is smaller than the similarity threshold T_i, ε is labeled as a "normal" signal.
3) Otherwise, is labeled as an "candidate" signal.

Usually, the "unimportant" patterns are very different from the patterns in the current reference dataset and "normal" structures are similar structures in low density area in ESOINN layer. A perfect reference dataset should have enough samples at the position of the transition states which are critical for non-equilibrium simulations. And these structures usually located at the inter-class area in ESOINN layer. So, we select the unknown structures in the ration of 1:3:1 for "normal" "candidate" and "unimportant" structure if the unknown structures from step 4 are more than the predefined maximum number of structures added per round.

(7) Select the best structural hyper-parameters for meta-NNs in the DP layer and go to step 2
(8) Get the final ESOINN-DP potential model.

2.6 The ESOINN-DP package

The ESOINN-DP method developed in this work has been released as an open-source package (*https://github.com/tongzhugroup/ESOINN-DP*). The software is totally implemented with Python and consisted of 4 modules: sampling engine, training module, task scheduling module and a web-based monitor module.

The sampling engine supports multi-scale MD simulation methods such as MM, QM/MM, full QM, NN/MM and full NN. The MM calculation was performed by calling the AmberTools 18 package, while the QM calculation is mainly achieved by calling the Gaussian 16 or DFTB+ software. Other QM packages can also be easily implemented. The task scheduling module uses the producer-consumer model to coordinate task allocation. The program supports the

remote invocation of the Linux computing cluster deployed with PBS and LSF task operating system for parallel DFT calculations and GPU accelerated training of neural networks. And a user-friendly monitoring module helps users analyze the diversity of datasets, visualize the structure of ESOINN layer (by projection on a 2D plane), monitor the training process and evaluate the reliability of MD simulations.

The main process of ESOINN-DP can be started via a single line of command:

esoinn-dp -i parm

where the argument *parm* is the name of input parameter files in *json* format that specify the user's demands. Details of parameters used in ESOINN-DP can be found in the supplementary material.

3 Results and Discussion

3.1 Automatic construction of NNPES for water clusters

To illustrate the performance of ESOINN-DP, we first used it to construct NNPES for water clusters containing different numbers of water molecules, and performed MD simulations. To save the computational cost, the reference dataset was labeled at the PM6 level and the training of each meta-NN stops when the root means square error (RMSE) of predicted and labeled atomic forces is less than 0.13 $eV/\text{Å}$ (3 $kcal/(mol \cdot \text{Å})$) in each round of concurrent learning.

The size of water clusters in this test is up to 9 water molecules. 2 meta-NNs were used to learn each pattern labeled by ESOINN. After 6 rounds of concurrent learning, the construction of the NNPES was completed. During each round of active learning, 9 parallel gas phase MD simulation with NNPES for water clusters with 1~9 water molecules were performed to sample structures respectively. The MD simulation was performed in a cavity with a radius of 8 Å to avoid the water molecules escaping from each other and the temperature was maintained at 300 K with the Anderson thermostat method. The MD simulation is extended round by round until the simulation time extended to 80ps and the error indicator χ_t of all structures is less than $2\delta_{Force}$. In each round of sampling, up to 1000 structures were extracted to update the reference dataset.

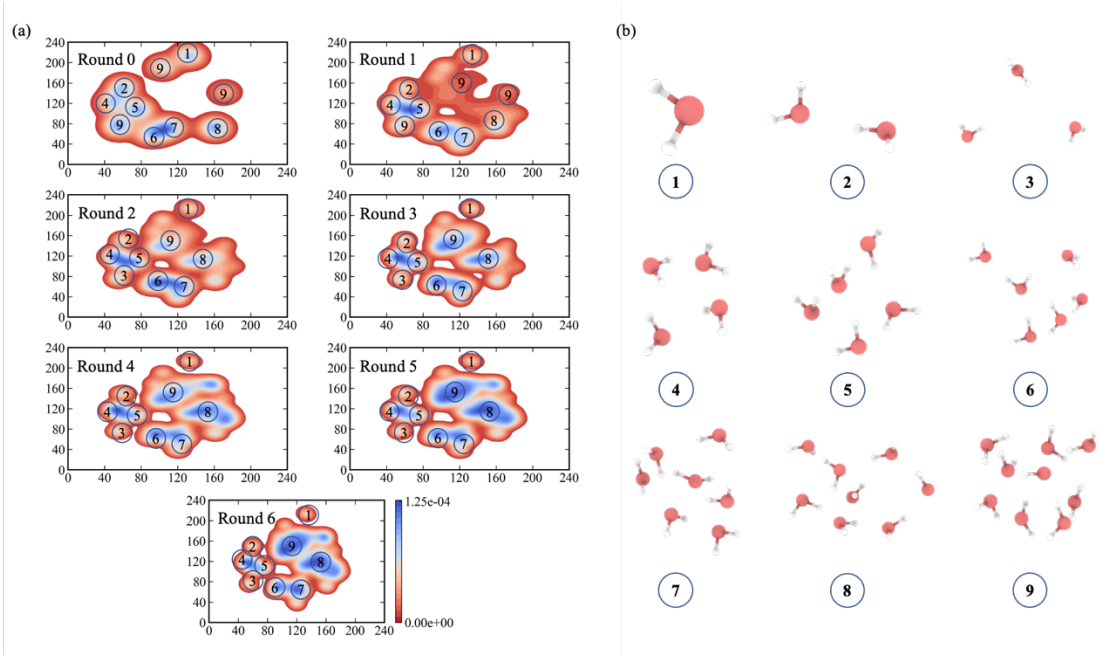

Figure. 5. (a), The distribution of structures in the reference dataset projected on the 2D plane with TSNE method and Sorted-EGCM descriptor in each concurrent learning process. (b), Representative structures.

As shown in Figure 5 and Table 1, starting from the initial dataset which contains only 9 structures corresponding to water cluster with 1~9 molecules, after the six rounds of sampling and updating progressed, different water structures were gradually added to the dataset. It can be seen that the systems with more water molecules have a larger proportion in the final reference dataset. The main reason is that the neural network potential model for systems with small degrees of freedom could be trained well with little reference data. It is clear that the redundancy of the dataset can be effectively reduced in ESOINN-DP.

The δ_{Force} is set to 0.18 $eV/\text{Å}$, which is calculated from the target accuracy of three meta-NNs:

$$\delta_{Force} = \sqrt{(\sum Target\ accuracy\ of\ each\ meta - NN(0.13\ eV/\text{Å}))^2} \quad (19)$$

In the process of constructing the ESOINN-DP model, the distribution of error indicators in each round are shown in Figure 6. It can be seen that the distribution of error indicator χ_t is gradually narrowing and smaller in the concurrent learning process. After the fourth round of active learning, the distribution of error indicator gradually converged. The time of MD simulation based on ESOINN-DP reached 80ps after 6 rounds and the error indicators of all structures were less than $2\delta_{Force}$.

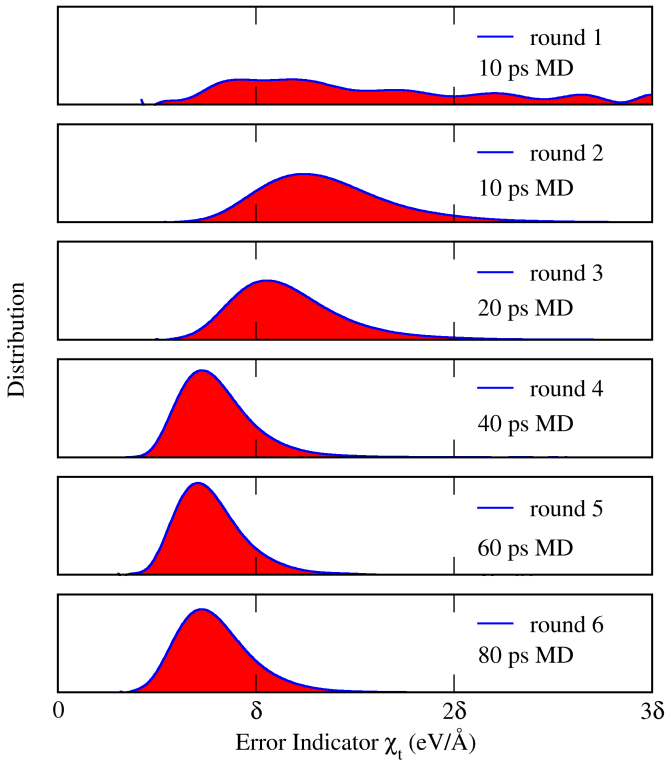

Figure. 6. Distribution of error indicator χ_t in 6 rounds of ESOINN-DP MD simulations for 1~9 waters in gas phase in active learning process. δ_{Force} was set to 0.18 $eV/$Å.

Table 1. The composition of reference dataset and the performance of ESOINN-DP model on training set and test set.

Number of water molecules	Training Set/Test Set		
	Structure Number	RMSE of E (eV)	RMSE of **F** (eV/Å))
1	78/20	0.02/0.02	0.02/0.03
2	78/20	0.04/0.04	0.05/0.05
3	110/27	0.05/0.04	0.05/0.05
4	263/66	0.06/0.07	0.12/0.12
5	264/66	0.08/0.08	0.12/0.12
6	366/91	0.13/0.14	0.15/0.15
7	589/145	0.12/0.13	0.14/0.16
8	1447/362	0.09/0.10	0.11/0.13
9	1734/433	0.11/0.12	0.11/0.12
Total	4929/1230	0.11/0.11	0.13/0.13

In addition, to further verify the reliability of NNPES, we extracted a series of frames from MD trajectories and then calculated their atomic forces using the PM6 method and compared them with the predicted values of NNPES. The results are shown in Figure 7. As can be seen, the root mean square error of atomic forces for all 4000 structures are smaller than 0.2 $eV/$Å. At the same time, it can be found that the error indicator χ_t is always greater than the RMSE between NNPES predictions and QM calculations, it means that the accuracy can be guaranteed with the constrains of error indicator. Thus, the error indicator χ_t can be used to evaluate the correctness of MD trajectories.

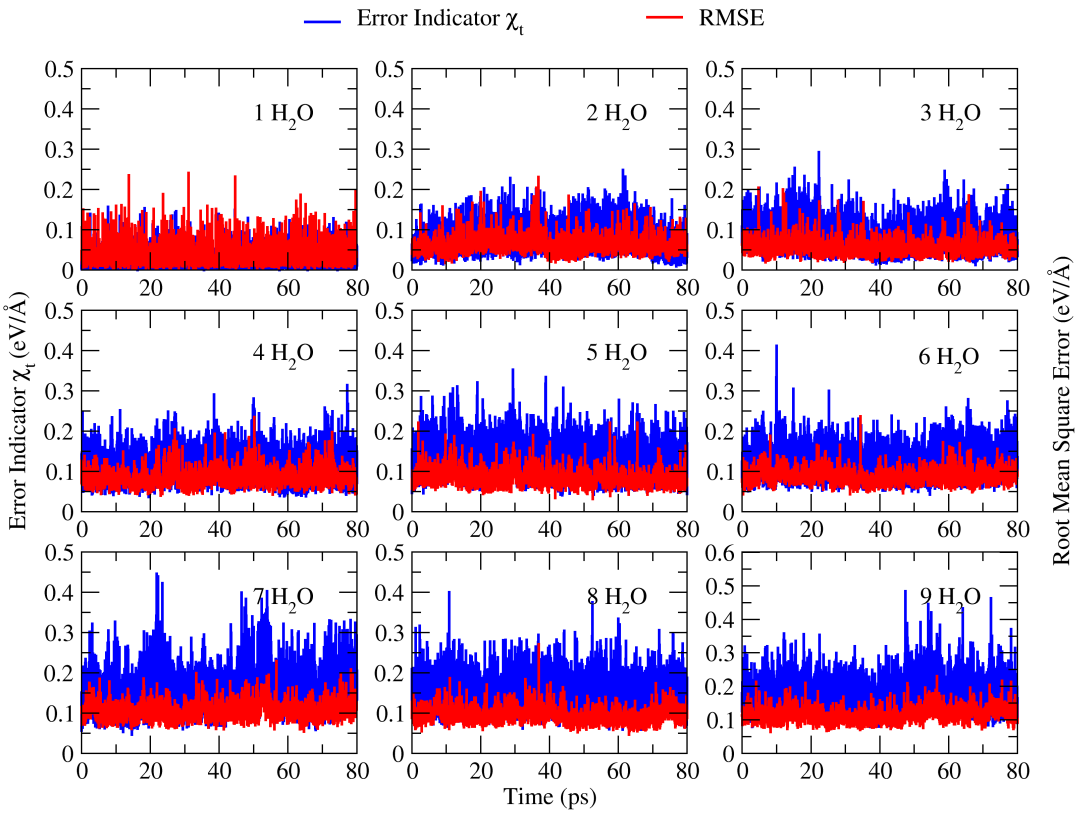

Figure 7. Time evolution of error indicator χ_t and root mean square error between ESOINN-DP predictions and PM6 calculations in 9 parallel MD simulations.

In each round of concurrent learning, the genetic algorithm discussed above was used to optimize of the structure of meta-NNs. Starts from the initial size of [200, 200, 200], the size of the smallest meta-NN was reduced to [71, 189, 218] after 6 rounds of concurrent learning. Which slightly improves the efficiency of prediction.

3.2 De-redundancy for a subset of the ANI-1 database

The performance of the NNPESs highly depends on the quality of the reference dataset. As mentioned above, this quality lies in two aspects. Firstly, the dataset must cover the target chemical space. Secondly, the dataset must be as small as possible so that the user can label it

at a higher QM level. To further test the performance of the ESOINN-DP method, we used it to remove the redundant data in a subset of the ANI-1 database. The ANI-1 dataset is one of the most famous open-source datasets for construct machine learning models for drug-like molecules, it contains the potential energy of more than 20M off equilibrium conformations for 57462 organic molecules.

To reduce the computational cost, we selected the first 6 million structures in the ANI-1 dataset and calculate their potential energy and atomic forces at the PM6 level. 20,000 structures were selected randomly from the 6 million structures as initial training set. In the first round of concurrent learning, an ESOINN-DP model was trained and used to predict the atomic forces of 800,000 randomly selected structures from the remaining data. The training of each meta-NN in the DP layer stops when the RMSE of atomic forces between predictions and labeled data are less than 0.2 $eV/\text{Å}$. In each round of the following concurrent learning process, 100,000 questionable and unknown structures were added to the reference dataset. And the test set was also keep increasing (Details can be found in Table 2). After 7 rounds of concurrent learning, the reference set consists of 60,0000 structures. At this point, no more than 1% of the data is considered as "unknown" by ESOINN-DP. This means that with the help of ESOINN-DP, we only need one tenth of the data (60,000) to cover the chemical space that was previously covered by 6 million data. This allows us to label these data at a much higher quantitative level.

It should be further noted that we are here only to demonstrate the capabilities of the ESOINN-DP method. The size of the final dataset can be further reduced if the number of rounds of concurrent learning is increased and the "questionable" and "unknown" data selected into the dataset in each round is reduced.

Table 2. The performance of the ESOINN-DP model in the de-redundancy process of a subset of ANI-1 database[a].

Iterations	Size of reference dataset	Size of the test set	δ $(eV/\text{Å})$	$\chi_t < \delta$	$\delta < \chi_t < 2\delta$	$2\delta < \chi_t < 3\delta$	RMSE[b] $(eV/\text{Å})$
1	20,000	800000	0.13	39.87%	50.75%	7.98%	0.82
2	100,000	2,000,000	0.21	58.83%	28.81%	7.74%	0.55
3	200,000	2,000,000	0.24	71.82%	24.10%	3.20%	0.33
4	300,000	2,000,000	0.25	89.93%	9.79%	0.24%	0.25
5	400,000	4,000,000	0.27	84.63%	12.62%	1.96%	0.24
6	500,000	5,500,000	0.25	73.43%	21.27%	3.57%	0.28
7	600,000	5,400,000	0.26	91.86%	7.28%	0.55%	0.20

[a] The QM calculations are performed at PM6 level.
[b] the RMSE is between predicted and labeled values in the dataset.

4 Conclusions

In this work, an automated neural network potential training method named enhanced self-organizing incremental neural network deep potential (ESOINN-DP) is proposed. With the help of ESOINN-DP, one can construct the reference datasets with little human intervention. More importantly, by the combination of ESOINN and the new proposed error indicator, this method ensures that the constructed dataset covers the target chemical space with minimum redundancy. In addition, ESOINN-DP can also optimize the hyper-parameters of neural networks automatically. The performance of the ESOINN-DP method has been well validated by develop neural network potential energy surfaces for water clusters and by de-redundancy of the ANI-1 database. The algorithms developed in this work have been integrated into an open-source software called ESOINN-DP (*https://github.com/tongzhugroup/ESOINN-DP*). It worth to mention that ESOINN defines not only the classification but also the boundaries of known data in an incremental learning manner, which means that we can further improve the efficiency of data sampling by using enhanced sampling algorithms such as metadynamics or deep molecular generation models under its guidance. This is the next step in the development of the method. We believe that the ESOINN-DP method provides a new idea for the construction of NNPES and especially, the reference datasets, and it can be used for MD simulations of various gas-phase and condensed-phase chemical systems.

Acknowledgment

This work was supported by the Ministry of Science and Technology of China (Grant No. 2016YFA0501700), the National Natural Science Foundation of China (Grants No. 21933010). We also thank the ECNU Multifunctional Platform for Innovation (No. 001) for providing supercomputer time.

Reference

1. T. B. Blank, S. D. Brown, A. W. Calhoun and D. J. Doren, *J. Chem. Phys.*, 1995, **103**, 4129-4137.
2. G. E. Moyano and M. A. Collins, *J. Chem. Phys.*, 2004, **121**, 9769-9775.
3. M. J. T. Jordan, K. C. Thompson and M. A. Collins, *J. Chem. Phys.*, 1995, **102**, 5647-5657.
4. K. C. Thompson, M. J. T. Jordan and M. A. Collins, *J. Chem. Phys.*, 1998, **108**, 8302-8316.
5. G. G. Maisuradze and D. L. Thompson, *J. Phys. Chem. A*, 2003, **107**, 7118-7124.
6. C. Qu, Q. Yu and J. M. Bowman, *Annu. Rev. Phys. Chem.*, 2018, **69**, 151-175.
7. J. Behler, S. Lorenz and K. Reuter, *J. Chem. Phys.*, 2007, **127**.
8. J. Behler and M. Parrinello, *Phys. Rev. Lett.*, 2007, **98**, 146401.
9. D. Lu, H. Wang, M. Chen, L. Lin, R. Car, W. E, W. Jia and L. Zhang, *Comput. Phys. Commun.*, 2021, **259**, 107624.
10. A. P. Bartok, M. C. Payne, R. Kondor and G. Csanyi, *Phys. Rev. Lett.*, 2010, **104**.
11. S. Chmiela, A. Tkatchenko, H. E. Sauceda, I. Poltavsky, K. T. Schutt and K. R. Muller, *Sci. Adv.*, 2017, **3**.
12. K. T. Schutt, F. Arbabzadah, S. Chmiela, K. R. Muller and A. Tkatchenko, *Nat Commun*, 2017, **8**.
13. X. Chen, M. S. Jorgensen, J. Li and B. Hammer, *J. Chem. Theory Comput.*, 2018, **14**, 3933-3942.
14. L. F. Zhang, J. Q. Han, H. Wang, R. Car and E. Weinan, *Phys. Rev. Lett.*, 2018, **120**.
15. K. Hansen, G. Montavon, F. Biegler, S. Fazli, M. Rupp, M. Scheffler, O. A. von Lilienfeld, A. Tkatchenko and K. R. Muller, *J. Chem. Theory Comput.*, 2013, **9**, 3404-3419.
16. J. Behler, *J. Chem. Phys.*, 2016, **145**.
17. J. Behler, *Angew Chem Int Edit*, 2017, **56**, 12828-12840.

18. T. Morawietz and J. Behler, *Z Phys Chem*, 2013, **227**, 1559-1581.
19. T. Morawietz and J. Behler, *J. Phys. Chem. A*, 2013, **117**, 7356-7366.
20. M. del Cueto, X. Zhou, L. Zhou, Y. Zhang, B. Jiang and H. Guo, *J. Phys. Chem. C*, 2020, **124**, 5174-5181.
21. C. Hu, Y. Zhang and B. Jiang, *J. Phys. Chem. C*, 2020, **124**, 23190-23199.
22. X. Lu, X. Wang, B. Fu and D. Zhang, *J. Phys. Chem. A*, 2019, **123**, 3969-3976.
23. H. Wang and W. Yang, *J. Chem. Theory Comput.*, 2019, **15**, 1409-1417.
24. H. Wang and W. T. Yang, *J. Phys. Chem. Lett.*, 2018, **9**, 3232-3240.
25. O. T. Unke, D. Koner, S. Patra, S. Käser and M. Meuwly, *Mach. Learn.: Sci. Technol.*, 2020, **1**, 013001.
26. A. Jonayat, A. C. Van Duin and M. J. Janik, *ACS Appl. Energy Mater.*, 2018, **1**, 6217-6226.
27. A. C. Rajan, A. Mishra, S. Satsangi, R. Vaish, H. Mizuseki, K.-R. Lee and A. K. Singh, *Chem. Mater.*, 2018, **30**, 4031-4038.
28. G. Pilania, J. E. Gubernatis and T. Lookman, *Comput. Mater. Sci*, 2017, **129**, 156-163.
29. R. Jinnouchi, F. Karsai and G. Kresse, *Phys. Rev. B*, 2019, **100**, 014105.
30. V. Botu, R. Batra, J. Chapman and R. Ramprasad, *J. Phys. Chem. C*, 2017, **121**, 511-522.
31. J. Behler, *Int. J. Quantum Chem*, 2015, **115**, 1032-1050.
32. H. E. Sauceda, S. Chmiela, I. Poltavsky, K. R. Muller and A. Tkatchenko, *J. Chem. Phys.*, 2019, **150**.
33. A. Singraber, T. Morawietz, J. Behler and C. Dellago, *J. Chem. Theory Comput.*, 2019, **15**, 3075-3092.
34. A. Bogdan, M. J. Molina, M. Kulmala, H. Tenhu and T. Loerting, *Proc Natl Acad Sci USA*, 2013, **110**, E2439.
35. A. S. Abbott, J. M. Turney, B. Zhang, D. G. A. Smith, D. Altarawy and H. F. Schaefer, *J. Chem. Theory Comput.*, 2019, **15**, 4386-4398.
36. E. Quintas-Sánchez and R. Dawes, *J. Chem. Inf. Model*, 2019, **59**, 262-271.
37. E. Quintas-Sánchez, R. Dawes, K. Lee and M. C. McCarthy, *J. Phys. Chem. A*, 2020, **124**, 4445-4454.
38. S. Venturi, R. L. Jaffe and M. Panesi, *J. Phys. Chem. A*, 2020, **124**, 5129-5146.
39. M. A. Collins, *Theoretical Chemistry Accounts*, 2002, **108**, 313-324.
40. L. M. Raff, M. Malshe, M. Hagan, D. I. Doughan, M. G. Rockley and R. Komanduri, *J. Chem. Phys.*, 2005, **122**.
41. J. Behler, *Physical Chemistry Chemical Physics*, 2011, **13**, 17930-17955.
42. L. F. Zhang, D. Y. Lin, H. Wang, R. Car and W. N. E, *Phys Rev Mater*, 2019, **3**.
43. W. Wang, T. Yang, W. H. Harris and R. Gomez-Bombarelli, *Chem Commun (Camb)*, 2020, **56**, 8920-8923.
44. Q. Lin, Y. Zhang, B. Zhao and B. Jiang, *J. Chem. Phys.*, 2020, **152**, 154104.
45. S. J. Ang, W. Wang, D. Schwalbe-Koda, S. Axelrod and R. Gómez-Bombarelli, *Chem*, 2021, DOI: https://doi.org/10.1016/j.chempr.2020.12.009.
46. Y. Zhang, H. Wang, W. Chen, J. Zeng, L. Zhang, H. Wang and W. E, *Comput. Phys. Commun.*, 2020, **253**, 107206.
47. T. T. Nguyen, E. Szekely, G. Imbalzano, J. Behler, G. Csanyi, M. Ceriotti, A. W. Gotz and F. Paesani, *J. Chem. Phys.*, 2018, **148**.
48. J. Behler, *J. Chem. Phys.*, 2011, **134**.
49. G. Imbalzano, A. Anelli, D. Giofre, S. Klees, J. Behler and M. Ceriotti, *J. Chem. Phys.*, 2018, **148**.
50. M. Rupp, A. Tkatchenko, K. R. Muller and O. A. von Lilienfeld, *Phys. Rev. Lett.*, 2012, **108**.
51. J. S. Smith, O. Isayev and A. E. Roitberg, *Chem. Sci.*, 2017, **8**, 3192-3203.
52. S. Furao, T. Ogura and O. Hasegawa, *Neural Networks*, 2007, **20**, 893-903.
53. H. W. Zhang, X. Xiao and O. Hasegawa, *Ieee T Neur Net Lear*, 2014, **25**, 1096-1105.