JCIM JOURNAL OF CHEMICAL INFORMATION AND MODELING

pubs.acs.org/jcim Article

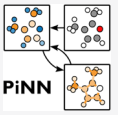
PiNN: A Python Library for Building Atomic Neural Networks of Molecules and Materials

Yunqi Shao, Matti Hellström,* Pavlin D. Mitev, Lisanne Knijff, and Chao Zhang*

Cite This: J. Chem. Inf. Model. 2020, 60, 1184−1193 Read Online

ACCESS | Metrics & More | Article Recommendations | Supporting Information

ABSTRACT: Atomic neural networks (ANNs) constitute a class of machine learning methods for predicting potential energy surfaces and physicochemical properties of molecules and materials. Despite many successes, developing interpretable ANN architectures and implementing existing ones efficiently are still challenging. This calls for reliable, general-purpose, and open-source codes. Here, we present a python library named PiNN as a solution toward this goal. In PiNN, we designed a new interpretable and high-performing graph convolutional neural network variant, PiNet, as well as implemented the established Behler−Parrinello neural network. These implementations were tested using datasets of isolated small molecules, crystalline materials, liquid water, and an aqueous alkaline electrolyte. PiNN comes with a visualizer called PiNNBoard to extract chemical insight "learned" by ANNs. It provides analytical stress tensor calculations and interfaces to both the atomic simulation environment and a development version of the Amsterdam Modeling Suite. Moreover, PiNN is highly modularized, which makes it useful not only as a standalone package but also as a chain of tools to develop and to implement novel ANNs. The code is distributed under a permissive BSD license and is freely accessible at https://github.com/Teoroo-CMC/PiNN/ with full documentation and tutorials.



INTRODUCTION

One major task of computational chemistry is to map the structure of a molecule or a material to its property, that is, $f: \{\vec{x}_i, Z_i\} \rightarrow P$. When P is the total energy, then the task is to devise computational methods to find approximate solutions to the Schrödinger equation, as Dirac foresaw in his 1929 account[1] and what generations of computational and theoretical chemists have been devoted to. What is even more challenging is to do the reverse $f: P \rightarrow \{\vec{x}_i, Z_i\}$, that is, to propose new structures that have properties of particular value.

To address these challenges, machine learning (ML) has attracted considerable attention and efforts in computational chemistry and materials discovery,[2−4] and many different types of ML methods have been successfully applied in those areas. In this work, we will focus on atomic neural networks (ANNs), that have been very successful in predicting physicochemical properties, approximating potential energy surfaces (PES),[5,6] and allowing for simulations of large-scale systems with the accuracy of reference electronic structure calculations but at only a fraction of the computational cost.

Despite this great promise and present success, the development of ANNs is not straightforward. ANNs must preserve rotational, translational, and permutational invariances of the system, which was recognized in the early days of ANN development.[7] Besides using functions of internal coordinates, which are rotationally and translationally invariant, a symmetrization process was proposed to preserve also the permutational invariance. However, such models could only be applied to systems of a given size and large-scale simulations were not possible.

Behler−Parrinello neural networks (BPNNs),[5] or high-dimensional neural network potentials, introduced the approach of partitioning the total potential energy of the system into effective atomic contributions. Not only does the atomic energy approach enable applying a trained ANN to systems of different sizes, but it also transforms the problem of describing the full system to that of describing the local chemical environment of each atom.[8,9] BPNNs have been successfully constructed for a wide range of molecules[10−13] and materials.[14−17]

The BPNN architecture relies on calculating fingerprints of the atomic environment using a set of symmetry functions that need to be selected before the fitting procedure can begin.[5] In contrast, the features are automatically learned via a feature hierarchy rather than handcrafted in convolutional neural networks, one of the most successful end-to-end techniques in the field of deep learning, which won the ImageNet competition in 2012.[18] Because molecules and materials can be viewed as fully connected graphs, this led to the development of graph convolution neural networks (GCNN) in atomic systems.[6,19,20] Hierarchical atomic features can be obtained by applying multistage concatenated convolution operations and this leads to impressive performance for a variety of systems.[6,21−26] Among those, SchNet[21] is a leading example for extending GCNN methods to the modeling of both molecules and materials.

Despite these progresses, developing interpretable ANN architectures,[27,28] and implementing existing ones efficiently

Received: October 27, 2019
Published: January 14, 2020

https://dx.doi.org/10.1021/acs.jcim.9b00994
J. Chem. Inf. Model. 2020, 60, 1184−1193

are still challenging. Therefore, to promote the application of ANNs in computational chemistry and materials science communities, reliable, general-purpose, and open-source codes are needed.[29−33] Here, we present a python library named PiNN as a solution toward this goal.

PiNN supports both BPNNs and GCNNs. The unique features of PiNN are, for example, (i) that it contains a new interpretable and high-performing GCNN variant, namely, PiNet and (ii) that the interpretation of GCNN in PiNN is given by visualizing feature activation using a user-friendly JavaScript plugin PiNNBoard, instead of doing dimension reduction of embedding vectors[22,25,26] or generating 3D potential map of a test particle.[21] Moreover, PiNN provides analytical stress tensor calculations for lattice optimizations and constant pressure MD simulations.

In the following, we first introduce PiNN's representation and abstraction of ANNs with focus on PiNet, an interpretable GCNN architecture we developed. Then, we discuss the implementation and package features of PiNN. After that, we present different case studies for molecules, crystalline materials, and liquids. Finally, we conclude with an outlook.

PINN'S REPRESENTATION OF ATOMIC NEURAL NETWORKS

Representing Local Chemical Environments. The many-body expansion decomposes the total potential energy E_{tot} of a system of N atoms into n-body terms $E^{(n)}$

$$\begin{aligned} E_{\text{tot}} &= \sum_{i}^{N} E^{(1)}(\vec{x}_i;\, Z_i) + \sum_{j>i}^{N} E^{(2)}(\vec{x}_i, \vec{x}_j;\, Z_i, Z_j) \\ &\quad + \sum_{j>i}^{N}\sum_{k>j}^{N} E^{(3)}(\vec{x}_i, \vec{x}_j, \vec{x}_k;\, Z_i, Z_j, Z_k) + \ldots \\ &= \sum_{i}^{N} E_i(\vec{x}_1, \ldots, \vec{x}_N;\, Z_1, \ldots, Z_N) \end{aligned} \tag{1}$$

where $\vec{x}_i$ is the atomic position and Z_i is the atomic number. In ANNs discussed in this work, E_{tot} by construction equals the sum of all effective atomic energies E_i. Note that one may also apply this construction to other atomic properties, such as partial charges.

In the construction of ANNs, descriptors that satisfy certain conditions such as symmetry invariance are needed. Two categories of descriptors have been developed to represent local chemical environments around atoms. One is inspired by the many-body expansion to include radial and angular terms[34] (but this does not mean only two-body and three-body information are captured), for example, atom-centered symmetry functions,[5,35] Faber−Christensen−Huang−Lilienfeld representation,[36] and local reference frames.[37] The other is using the expansion of atomic density in terms of orthogonal radial functions and spherical harmonics,[38] for example, smooth overlap of atomic positions.[39] However, as pointed out recently,[38,40] these phenomenological categories are indeed connected and one can systematically build up many-body representations from atomic cluster expansion in terms of single-bond basis functions,[40] tensor product of symmetrized two-body dyad,[38] or finite powers of the two-body base kernel.[41]

The key insight of these recent works is that the full set of bond vectors originating from a given center atom in an atomic configuration of molecules or materials contains the complete information about the corresponding structure, and a many-body representation can be generated through interactions of bond vectors. In fact, the same inspiration is shared in GCNNs, where the starting point is a directed graph of the structure with annotated bonds as edges and the many-body representation in the latent space is generated from a series of graph convolution blocks (see Pairwise Interactions and Interaction Pooling for more discussions).

Behler−Parrinello and Graph Convolutional Neural Networks. Here, we will briefly describe two important ANN architectures relevant to PiNN.

In BPNNs, the chemical environment around an atom is represented by a vector of symmetry functions.[5,8] Such symmetry functions are rotationally, translationally, and permutationally invariant and can capture both radial and angular features of the chemical environment within a cutoff radius R_c. Each chemical element has its own set of symmetry functions and also its own neural network architecture and fitted parameters. The adoption of the atomic energy approach (eq 1) in BPNNs makes the resulting ANN applicable to systems with an arbitrary number of atoms.

A GCNN[6,19,20] is a combination of a graph neural network and a convolutional neural network. A graph neural network considers the atoms as nodes and the pairwise interactions between them as weighted edges. Node and edge feature vectors are iteratively updated through, for example, a message passing function.[42] The ingredient of convolution is often recognized as a learnable radial filter that gathers information from neighboring atoms within a cutoff radius R_c and creates a feature hierarchy. Because the generation of node features includes the element specificity by construction (see eqs 2 and 3 in the next section), GCNN has exactly the same subnet for each element.

Pairwise Interaction and Interaction Pooling. In PiNN, we describe both BPNN and GCNN with two abstractions (Figure 1): pairwise interaction operation (PI) and interaction pooling operation (IP). We start by labeling atom i with a set of scalar numbers or an atomic property $\vec{P}_i$. It is sufficient to use the nuclear charge Z_i or a one-hot embedding of the element as the starting point. Then, we create the pairwise interaction $\vec{I}_{ij}$ as a function of the initial atomic properties of two atoms and their distance r_{ij} (eq 2).

$$\vec{I}_{ij}^{\,t} = \text{PI}(\vec{P}_i^{\,t}, \vec{P}_j^{\,t}, r_{ij}) \tag{2}$$

where t is an iterator. For BPNN, $t = 0$ and for GCNN, $t + 1$ goes up to the number of graph convolution (GC) blocks (Figure 1).

The opposite of the PI operation is IP. Namely, this operation creates an atomic property from all the pairwise interactions associated with that atom. This is done by passing the summation over all the pairwise interactions to another function called IP (eq 3). The summation ensures the permutation invariance of the generated atomic property.

$$\vec{P}_i^{\,t+1} = \text{IP}\left(\sum_j \vec{I}_{ij}^{\,t}\right) \tag{3}$$

The combination of the PI and IP operations essentially creates an updated atomic property, with information collected from neighboring atoms. This is referred to as atomic fingerprints, continuous-filter convolution, or neural message

https://dx.doi.org/10.1021/acs.jcim.9b00994
J. Chem. Inf. Model. 2020, 60, 1184−1193

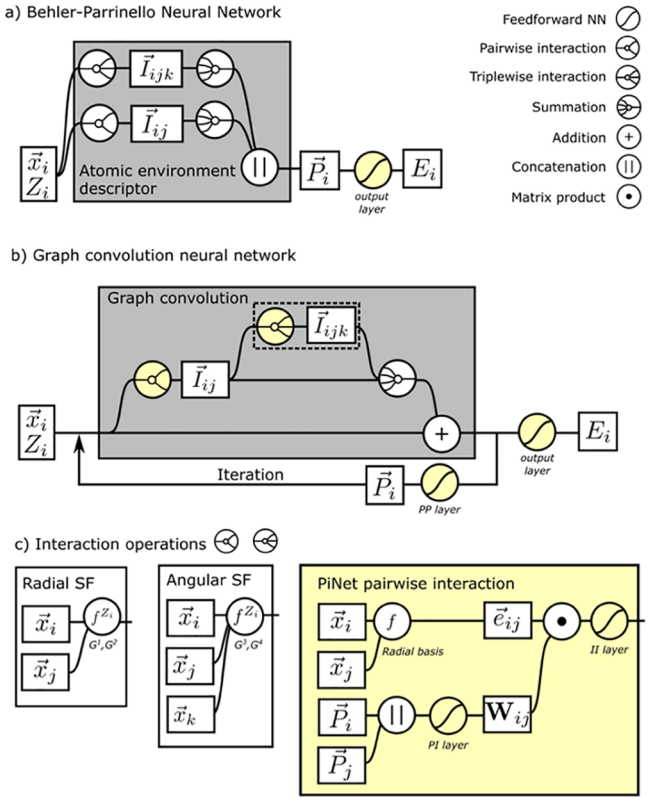


Figure 1. Illustration of the abstractions of BPNN and GCNN frameworks in PiNN for calculating atomic energies E_i (eq 1). The operations containing trainable variables are filled with yellow color. Operations inside the dashed box are not yet implemented in PiNN but extendable. See "Pairwise Interaction and Interaction Pooling" for more explanations.

passing in the literature.[9,21,42] Through multiple GC blocks PI + IP → PI + IP..., I_{ij} also gets updated in this process. The actual forms of the IP and PI functions are different in each ANN and this gives the freedom to create novel architectures (Figure 1).

To illustrate the many-body representation in GCNN, we use a single water molecule as an example to show how three-body interactions are generated from two-body representation, as shown in Figure 2. In the stage of embedding, the water molecule turns into a directed graph with node feature $\vec{P}_i^t$ and edge feature $\vec{I}_{ij}^t$ for i = 1,2,3 and j = 1,2,3. It is worth noting that $\vec{I}_{ij}^t$ is a high-dimensional feature vector and $\vec{I}_{ij}^t$ is not necessarily be the same as $\vec{I}_{ji}^t$. At t = 0, the PI operation will take node features $\vec{P}_i^0$, $\vec{P}_j^0$, and the bond r_{ij} to generate the interaction $\vec{I}_{ij}^1$. Subsequently, the IP operation will update the node feature $\vec{P}_i^1$ by summing all interactions centered at atom i. $\vec{P}_i^1$ is equivalent to a vector of radial symmetry function values in BPNNs for atom i. At t = 1, the PI operation generates a new interaction $\vec{I}_{ij}^1$ by repeating the same procedure, which is followed by another IP operation. What is important is to note that $\vec{P}_i^2$ is not only a three-body function but also a unique representation of the local chemical environment of atom i. Moreover, the pairwise interaction $\vec{I}_{ij}^1$ is already a three-body function in GCNN, and the inclusion of explicit triplewise interactions $\vec{I}_{ijk}$ is possible but unnecessary.

To close this section, we note that although multiple GC blocks are capable of generating multibody representations, there is no general proof regarding the uniqueness of GCNN

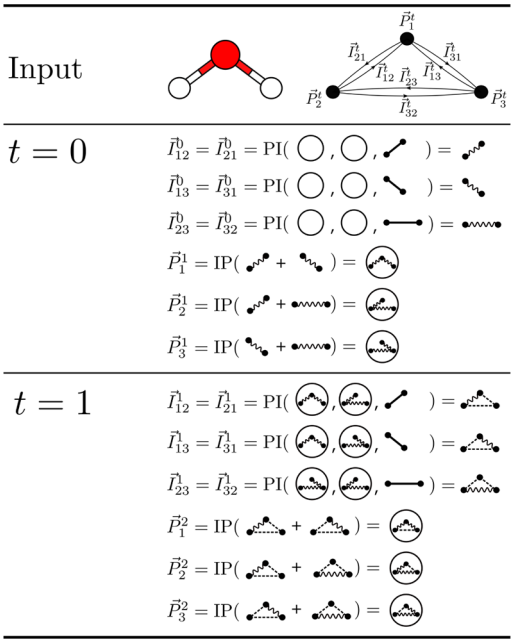


Figure 2. Illustrating how three-body representations for a water molecule are generated from two-body representations and graph convolution blocks in GCNN. $\vec{I}_{ij}^t$ and $\vec{P}_i^t$ are the pairwise interaction between atom i and j and the node feature vector of atom i, respectively. t is the iterator in graph convolution blocks.

representations. To the authors' knowledge, the closest example shows that GCNNs can be as powerful as the Weisfeiler−Lehman algorithm in detecting isomorphic graphs.[43]

Architecture of PiNet. With the formalism of PI and IP, we designed and implemented a new GCNN variant called PiNet. This network is motivated by the aim to make the activations (pairwise interactions and atomic properties) of ANNs interpretable.

The main idea behind PiNet is to define the pairwise interaction as a function of distance whose exact form depends on the atomic properties of both interacting atoms. In other words, the weight matrix W_{ij} used for generating the pairwise interaction $\vec{I}_{ij}$ depends on both atomic properties $\vec{P}_i$ and $\vec{P}_j$. This makes each component of the pairwise interaction $\vec{I}_{ij}$ have different radial dependence, which is unique in PiNet and differs from the common approach of using a single radial-dependent filter function (attention mask).[21,23,24]

In PiNet, the PI operation is split into three steps (Figure 1c): (i) expressing the interatomic distances in a radial basis $\vec{e}_{ij}$; (ii) activation through the PI "layer", which is a feed-forward NN generating a weight matrix W_{ij} from the atomic properties $\vec{P}_i$ and $\vec{P}_j$; and (iii) activation through the II (Interaction to Interaction) layer, which is a feed-forward NN using the information from the previous two steps as input and generating the interaction $\vec{I}_{ij}$.

The radial basis $\vec{e}_{ij}$ for the pair of atoms i and j is calculated as n_{basis} Gaussian functions multiplied by a cutoff function f_c.

$$\vec{e}_{ij} = f_c(r_{ij}) \cdot [e^{-\eta(r_{ij}-r_1)^2}, e^{-\eta(r_{ij}-r_2)^2}, ...] \quad (4)$$

The centers of the Gaussian functions r_1, r_2, ..., $r_{n_{\mathrm{basis}}}$ are chosen to be evenly spaced between 0 and the cutoff radius R_c, and the hyperparameter η determines the width of the Gaussian functions.

https://dx.doi.org/10.1021/acs.jcim.9b00994
J. Chem. Inf. Model. 2020, 60, 1184−1193

We have chosen the cutoff function f_c given in ref 8, which ensures that the interaction and its gradient vanish at the cutoff radius R_c

$$f_c(r_{ij}) = \begin{cases} 0.5 \cdot \left[\cos\left(\frac{\pi r_{ij}}{R_c}\right) + 1\right] & \text{for } r_{ij} < R_c \\ 0 & \text{for } r_{ij} \geq R_c \end{cases} \quad (5)$$

The PI layer generates a weight matrix W_{ij} from the concatenated atomic properties $\vec{P}_i$ and $\vec{P}_j$

$$W_{ij} = \mathrm{NN}^{\text{PI-layer}}([\vec{P}_i, \vec{P}_j]) \quad (6)$$

The II layer calculates $\vec{I}_{ij}$ by means of a different feed-forward NN

$$\vec{I}_{ij} = \mathrm{NN}^{\text{II-layer}}(W_{ij}\vec{e}_{ij}) \quad (7)$$

The radial basis $\vec{e}_{ij}$ decays to zero beyond the cutoff radius and all biases are set to zero in the II layer, which guarantees the smoothness of PES in PiNet in contrast to other approaches where the interaction is directly generated from the distance and the atomic properties.[22,25]

It is worth mentioning that despite the fact that PiNet does not include a triplewise filter $\vec{I}_{ijk}$ in the current implementation, angular information is nonetheless captured through the iteration of the GC block.

After the PI operation, the updated atomic property $\vec{P}_i$ is calculated from all pairwise interactions $\vec{I}_{ij}$ as part of an IP operation (eq 3). Before passing this atomic property on to the next GC block in the iteration, a PP (Property-to-Property) layer (another feed-forward NN) is used for further refinement (see Figure 1b).

IMPLEMENTATION AND PACKAGE FEATURES

Modularized ANN. PiNN is modularized so as to cater to the need of different uses. The task of building an ANN is split into three stages: dataset preparation, ANN definition, and model definition. As shown in Figure 3, PiNN primarily

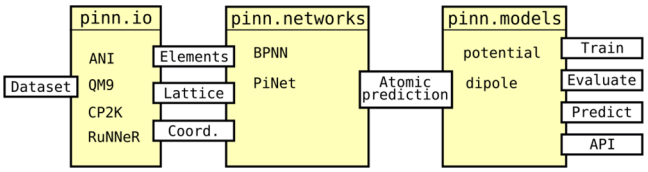


Figure 3. Illustration of the modularized structures in PiNN.

consists of three modules: input/output (io), networks, and models, which correspond to these three stages. The network does not specify the physical meaning of atomic predictions. That is instead defined through the loss function in a model. The model also contains hyperparameters such as the learning rate and the regularization.

This design enables users to easily import an arbitrary dataset without touching the code base. Similarly, researchers interested in implementing a new ANN architecture could implement it as one of the "networks" and use the rest of the modules to test its performance. Finally, the "models" module could be extended to use the existing networks for different property predictions (e.g., dipole moments and partial charges in Case Studies) or to interface with other external codes.

Notably, this modularized structure also decouples the implementation of common neural network building blocks, such as activation functions and optimization algorithms, from the ANN architecture. For example, a number of activation functions (e.g., tanh, logistic, or softplus) can be chosen in PiNN as adjustable hyperparameters of the network architecture.

Dataset Preparation. PiNN is implemented on top of TensorFlow's estimator API,[44] which requires the training data to be represented with the dataset class.

To enable easier utilization of data from different sources, we provide the functionality of creating custom dataset loaders. Given a list of data files and a reader function, the dataset loader can be used to split the dataset, as is commonly required for neural network training. Similar procedures can be applied to trajectory files or databases. Further instructions for building datasets are provided in the documentation.[45]

Several types of dataset loaders are provided, such as for the QM9 dataset,[46] ANI-1 dataset,[47] Numpy[48] formatted datasets, ASE[49] databases, RuNNer-format[9,50] datasets, and CP2K[51] trajectories in XYZ format.

In addition, the dataset objects can be saved into TensorFlow's TFRecord file format, for fast reading and serving the dataset from a remote file system.

Interfaces with ASE and AMS. PiNN comes with interfaces connecting the trained neural network potential to ASE[49] through its calculator class (see Figure 4) and to a development version of AMS[52] as an external engine.

```
from pinn.models import potential_model
from pinn.calculator import PiNN_calc
from ase.io import read
calc = PiNN_calc(
    potential_model('/path/to/model/'))
calc.calculate(read('molecule.xyz'))
```

Figure 4. Code example for using a trained model as an ASE calculator.

Periodic boundary conditions are seamlessly supported. The neighbor list and pairwise distances of the periodic system are efficiently calculated using a cell list algorithm[53] that we implemented in TensorFlow.

With either ASE or AMS, the PiNN implementations of BPNNs and PiNet can be used to run, for example, geometry optimizations and molecular dynamics (MD) simulations of both gas-phase and condensed-phase systems. The simulation trajectories can be visualized using their respective graphical user interfaces.

Moreover, the implementation of PiNN as an ASE calculator allows it to be interfaced easily with other codes. For example, the path integral molecular dynamics (PIMD) code i-PI[54] can communicate with ASE calculators via a socket protocol, allowing PIMD simulations to be run with PiNN energies and forces.

Pressure Calculations. Computing the stress tensor requires attention, especially when the potential is not pairwise-additive.[55] Although an ANN potential does not look like pairwise-additive, the stress tensor calculated using the pairwise form $\vec{F}_{ij} \cdot \vec{r}_{ij}$ yields the same result as the atomic form given in ref 55. These results were also validated using the finite difference of the potential energy with respect to the

https://dx.doi.org/10.1021/acs.jcim.9b00994
J. Chem. Inf. Model. 2020, 60, 1184−1193

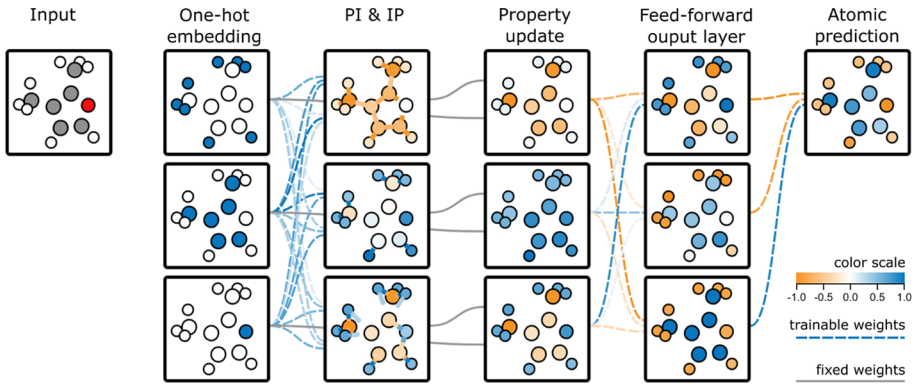


Figure 5. Visualization of a minimalistic PiNet for a 2,3-dimethylfuran molecule generated directly with PiNNboard. For simplicity, only one graph convolution block without PP and II layers was used. The atoms and bonds in each box indicate the normalized activations of the atomic property and pairwise interaction, respectively. Dashed lines show the normalized trainable weights. Activation of pairwise interactions with an absolute value smaller than 0.3 is not shown for the sake of clarity.

change of the volume. Therefore, the instantaneous pressure $P(t)$ during an MD simulation is calculated as

$$P = \frac{1}{3\Omega}\left(\sum_{i}^{N}\frac{|\vec{p}_i|^2}{m_i} + \sum_{i}^{N}\sum_{j>i}^{N}\vec{F}_{ij}\cdot\vec{r}_{ij}\right) \quad (8)$$

where $\vec{F}_{ij} = \partial E_{tot}/\partial\vec{r}_{ij}$ is the derivative of the potential energy with respect to the pairwise displacement vectors $\vec{r}_{ij}$,[56] $\vec{p}_i$ is the momentum of atom i, m_i is the mass of that atom, and Ω is the system volume. The first term on the right hand side of the equation is the ideal gas contribution, and the second term is the virial pressure output from PiNN.

Visualization of Atomic Neural Networks with PiNNboard. It has been shown that a visualization of the activations can provide insights into their functions in convolutional neural networks.[57] Such a visualization can also serve as a diagnostic tool to inspect and improve network architectures. Therefore, we developed a tool, PiNNboard, to visualize the activations of ANNs in atomic or pair forms. PiNNboard was implemented as a plugin for Tensorboard, TensorFlow's visualization toolkit.[44]

Here, we demonstrate PiNNboard with the minimalistic PiNet, trained on a subset of the QM9 dataset containing 50,604 organic molecules consisting of only C, H, and O atoms. The network was trained for 3 million steps on the internal energy at 0 K (U_0), and details about the training setups can be found in the Case Studies section below.

In Figure 5, the activations of this minimalistic PiNet for the testing molecule 2,3-dimethylfuran are visualized using PiNNboard. Colored atoms indicate the contributions of each individual atom to atomic properties. Colored bonds between atoms indicate the pairwise interactions whose contributions are significant. Indeed, most of the interactions identified by the PI layer in PiNet can be recognized as covalent bonds in Figure 5. Interestingly, strong activations in the pairwise interaction are also observed between hydrogen atoms and their second neighbors in the case of the methyl groups.

Notably, three independently trained networks (Figure 6) reach quite different atomic energy partitions as well as different pairwise interactions for the testing molecules, despite their identical network structure, hyperparameters, and similar MAE. Therefore, one needs to be careful when interpreting atomic energies extracted from ANNs, which has also been pointed out recently.[58]

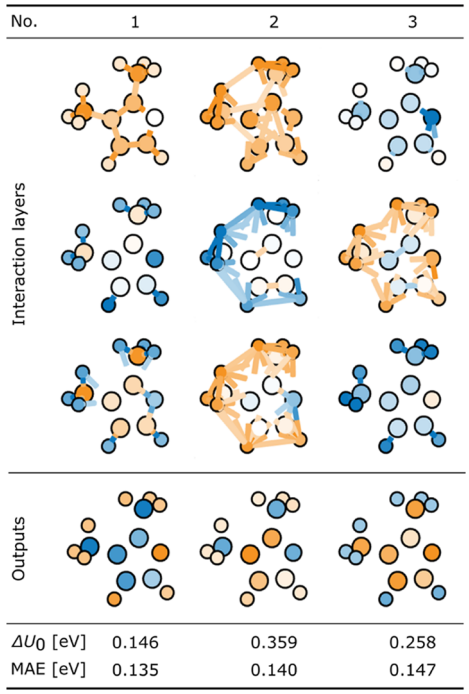


Figure 6. Activations of graph convolution and output layers after three independent trainings of the minimalistic PiNet shown in Figure 5. The prediction error in the U_0 of the testing molecule and the mean absolute error (MAE) on the validation set are also listed.

We also notice that the trained network (No.1 in Figure 6) which has the smallest prediction error for the testing molecule also provides the best chemical interpretability. This suggests that not only is PiNet capable of providing a state-of-the-art performance but also that the good outcome from PiNet can be rationalized in a chemically intuitive manner.

CASE STUDIES

For all discussed benchmarks, a network with five graph convolution (GC) blocks is used. The parameters are given in Table 1. Hyperbolic tangent activation functions were used in all layers, except for the output layer where a linear activation function is used for output.

Unless otherwise stated, an 80:20 dataset splitting was used for case studies, which means that 80% of the structures of

https://dx.doi.org/10.1021/acs.jcim.9b00994
J. Chem. Inf. Model. 2020, 60, 1184−1193

Table 1. Network Parameters Used in Case Studies

layer	architecture[a]	parameter	value
PI	[64]×10[b]	R_c	4.5 Å
II	[64, 64, 64, 64]	GC blocks	5
PP	[64, 64, 64, 64]	n_{basis}	10
output	[64, 1]	η	3.0 Å^{-2}

[a]The layer architecture is denoted with the number of nodes in the hidden layers and in the output layer. [64, 64, 64, 64] means a neural network with three hidden layers, each with 64 nodes and 64 output nodes. [b]The PI layer does not contain any hidden layer and the output dimension of the PI layer equals to the number of elements in W_{ij}.

each dataset were randomly chosen to train the neural network, and the remaining 20% were used for validation. The Adam[59] optimizer in TensorFlow[44] was used for gradient descent updates with a batch size of 100 samples; the training rate was set to 0.0003 and decayed by a factor of 0.994 every 100,000 steps, and other parameters were kept unchanged. A gradient norm clipping strategy was employed to avoid exploding gradient problems.[60] The trainings were terminated after 1−3 million gradient descent steps, which typically takes a day with a single NVIDIA TITAN V GPU card.

Because the purpose of PiNN is not to serve as a singular ANN architecture but to be used as a reliable and general-purpose library for further developments, we will not compare our results exhaustively with all other GCNN variants.[6,21−26] Instead, SchNet,[21] which pioneered the application of GCNN for modeling both molecules and materials, will be quoted as the main reference to put our results into perspective.

QM9 Dataset. The QM9 dataset[46] is a dataset made up of 134 k small organic molecules containing computed electronic, energetic, and thermodynamic properties at the B3LYP/6-31G(2df,p) level of theory,[61−63] which is often used for benchmarking ANNs. As commonly done in the literature, 30,054 structures that failed a consistency check were excluded during training and evaluation.[21,24]

PiNet reaches an MAE of 0.012 eV for the prediction of internal energy at 0 K, in comparison with 0.014 eV from SchNet. It is worth mentioning that the so-called chemical accuracy from thermochemistry measurements is about 0.043 eV.

As an example of property predictions, we used PiNet to predict partial charges by regressing only the molecular dipole moment μ

$$\mu = \left| \sum_{i=1}^{N} \tilde{q}_i \vec{r}_i \right| \tag{9}$$

where $\tilde{q}_i$ is the predicted partial charge on atom i. To ensure that the predicted total charge of each molecule is zero, we added a constraint term to the loss function.

By implementing this dipole model in PiNet, we predicted the dipole moment with an MAE of 0.018 D for the QM9 dataset. The network used to predict the dipole for the QM9 dataset was trained with a learning rate of 0.0001 and batch size of 200 structures instead.

To further validate the PiNet dipole model, we also calculated partial charges using the CM5 charge model, which has been parameterized to reproduce dipole moments from experiments or high-level quantum mechanical calculations.[64] The CM5 charges were calculated at the B3LYP/6-31G(2df,p) level of theory using the Gaussian package,[65] matching the original conditions in which the QM9 dataset was generated. Note that CM5 charges were not used in the training of PiNet. When comparing predicted partial charges from PiNet with those from CM5,[64] a good correlation was found, as shown in Figure 7, indicating that PiNet can generate

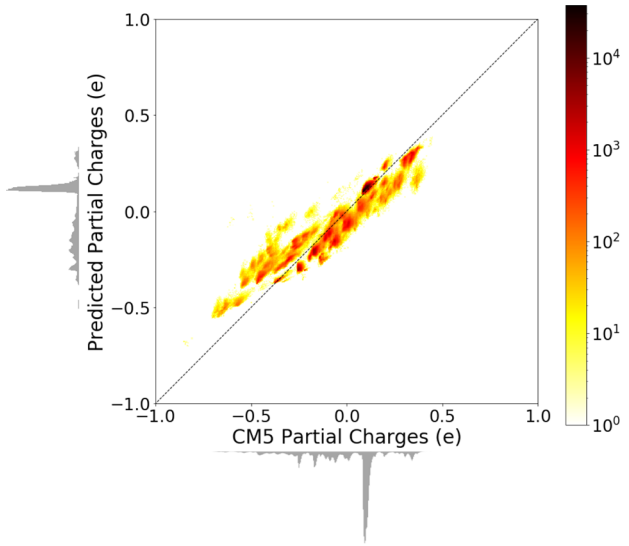


Figure 7. Correlation of the predicted partial charges from PiNet with those calculated from CM5 using the QM9 dataset.

physically meaningful partial charges. This result is particularly encouraging in light of the fact that only the scalar dipole moment was used during the fitting.[66]

Materials Project and Perovskite Datasets. We used the dataset ("MP-crystals-2018.6.1") provided by MEGNet,[25] which contains DFT-computed energies and band gaps for 69,640 crystals extracted from the Materials Project.[67] Training of PiNet was done with 60,000 crystal structures from this dataset. The trained PiNet leads to an MAE of 0.029 eV/atom for the prediction of formation energy on the test configurations, in comparison with that of 0.035 eV/atom from SchNet using the same number of structures in the training set. To put these numbers into perspective, we note that the accuracy of experimental measurement of formation energies is about 0.082 eV/atom.[68] Thus, PiNet provides also a subchemical accuracy for material modeling.

In addition, PiNet was benchmarked on a dataset consisting of 18,928 perovskite structures published by Castelli et al.[69] The achieved MAE of the total energy respective to the convex hull (for the purpose of assessing the thermodynamics metastability) from the trained PiNet is 0.042 eV/atom with a 60:40 splitting of the dataset. This is in comparison with the same MAE obtained from the crystal graph convolutional neural network (CGCNN) with an 80:20 splitting instead.[70] The learning curve of PiNet with this dataset is shown in Figure 8 (in logarithm scale).

Liquid Water Dataset. Here, we showcase the application of the BPNN implemented in PiNN to liquid water using the dataset published by Morawietz and Behler[71] based on the BLYP functional.[62,72] To facilitate the training, we also augmented this dataset using the original BPNN implementation[71] (see the Supporting Information for details). The set of symmetry functions was chosen to match the original ones;[16] however, hyperparameters such as the learning rate and

https://dx.doi.org/10.1021/acs.jcim.9b00994
J. Chem. Inf. Model. 2020, 60, 1184−1193

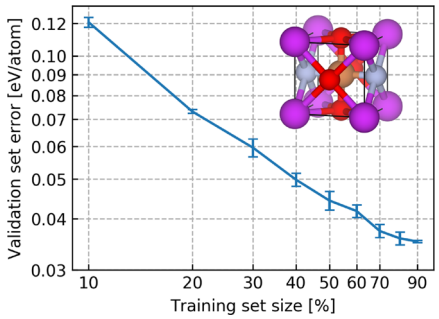


Figure 8. MAE of predicted energy above the hull from PiNet as a function of training configurations' size in the perovskites dataset.

optimizer are specific in PiNN. MD simulations were performed with the Berendsen thermostat[73] implemented in ASE and a patched Berendsen barostat (to include the missing ideal gas contribution in eq 8; for details, see ref 45).

After the training, the BPNN implemented in PiNN reaches a root mean squared error (RMSE) of 7 meV/H_2O for energy and 60 meV/Å for force components. These can be compared to 2 meV/H_2O and 70 meV/Å for energy and force, respectively, reported in ref 16. To validate this BPNN potential, we further carried out ab initio molecular dynamics (AIMD) simulations of the liquid water system at both NVT (constant particle, volume, and temperature) and NPT (constant particle, pressure, and temperature) ensembles with CP2K[51] and the BLYP functional.[62,72] Details of AIMD simulations can be found in the Supporting Information.

As shown in Figure 9, the BPNN potential generated with PiNN reproduces well the structure of liquid water from

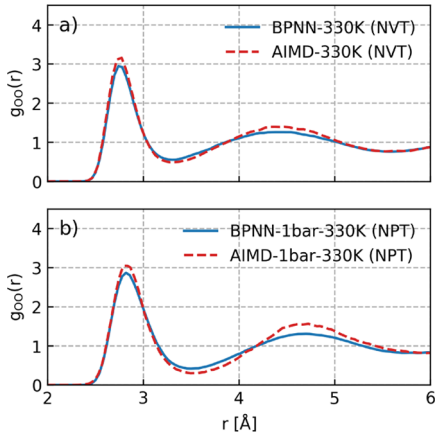


Figure 9. Calculated oxygen−oxygen radial distribution function of liquid water at (a) 330 K (NVT) and (b) 330 K and 1 bar (NPT) using BPNN implemented in PiNN and AIMD. The level of theory is BLYP.

AIMD simulations, particularly in the NVT ensemble. It is found that at 330 K and 1 bar, this BPNN potential predicts a density of 0.74(2) g/mL, in good agreement with 0.79(2) g/mL from AIMD simulations. Note that the AIMD simulations shown in Figure 9 were not used in the training and merely served the purpose of the cross-validation.

Proton Transfer Reactions. Finally, we demonstrate that PiNet can be applied to reactive MD simulations in which covalent bonds break and form. In particular, we take the example of proton transfer reactions in aqueous NaOH solutions.

We reused and slightly modified the dataset for NaOH(aq) solutions from ref 74, generated with the RPBE density functional[75] and Grimme's D3 dispersion correction.[76] This dataset was originally constructed for use with BPNN and the resulting BPNN potential was tested for various thermodynamic and dynamical properties of NaOH solutions.[77,78]

In this case, we used smaller layers in PiNet (with 16 nodes per layer rather than 64) as compared to the other case studies to prevent overfitting and optimized the parameters primarily to minimize the error in the predicted forces. We obtained an excellent RMSE of 0.11 eV/Å for the force components for both the training and validation sets, indicating that the fit did not suffer from overfitting.

Environment-dependent proton transfer free energy profiles for NaOH solutions were calculated with PiNet (Figure 10).

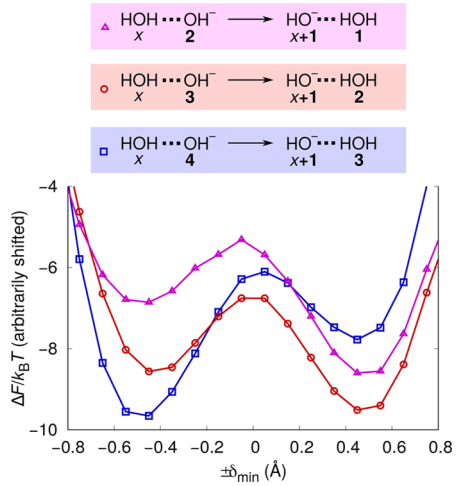


Figure 10. PiNet-calculated proton transfer free energy profile in 2.6 mol/L NaOH(aq) for OH^- and H_2O accepting different number of hydrogen bonds. The numbers beneath the molecules in the legend at the top refer to the total number of accepted hydrogen bonds. Here, we used a common definition of a hydrogen bond for which the O−O distance is smaller than 3.5 Å and the hydrogen-bonding angle is smaller than 30 degrees.

For each OH^-, the proton transfer coordinate δ_{min} is calculated as the difference in length between the hydrogen bond along which the proton is transferred and the covalent O−H distance for the bond that becomes broken.[79] Corresponding equilibrium MD simulations were run using an interface with the Amsterdam Modeling Suite (AMS),[52] and technical settings of MD simulations were chosen to be the same as in ref 74 for the sake of comparison.

Figure 10 illustrates how the free-energy landscape for proton transfer depends on the local hydrogen-bonding environments around OH^- and H_2O. Just as revealed in the previous work using BPNNs[74] and ab initio simulations,[79] proton transfer occurs predominately via a presolvation mechanism: OH^- mostly accepts four hydrogen bonds in its equilibrium structure, but the forward PT barrier is quite high in this case (blue curve). Instead, if via a hydrogen bond fluctuation, the OH^- only accepts three hydrogen bonds (red curve), and the forward PT barrier is much smaller.

Figure 11 shows the time required to evaluate the energy and forces for different system sizes of liquid water (averaged

https://dx.doi.org/10.1021/acs.jcim.9b00994
J. Chem. Inf. Model. 2020, 60, 1184−1193

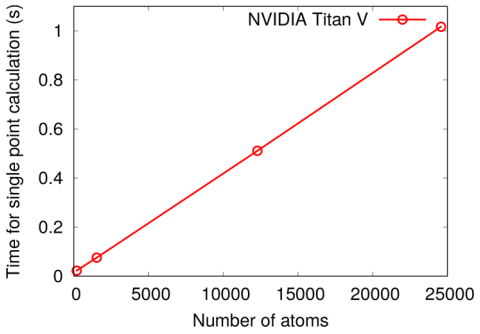


Figure 11. Computational cost as a function of system size for the NaOH PiNet model with a single GPU card.

over 100 samples), using the PiNet model parameterized for NaOH. This highlights one of the appealing features of ANNs in which the computational cost grows linearly with the number of atoms.

CONCLUSIONS

Here, we present PiNN, an open-source Python library for building ANNs of molecules and materials. In the current version of PiNN, BPNN and our GCNN variant PiNet have been implemented and benchmarked against several publicly available datasets as well as in-house data.

Built with the TensorFlow framework, PiNN allows for fast training of ANNs with GPUs. PiNN interfaces with ASE and AMS and provides several useful package features such as analytical stress tensor calculations and a visualizer of trained ANNs called PiNNBoard.

With modularized building blocks, PiNN can be used not only as a standalone package but also as a chain of tools to develop novel ANNs. In this work, we showed how such ANNs can be used for approximating potential energy surfaces, allowing for fast and accurate reactive MD simulations or for directly predicting several different physicochemical properties of molecules and materials, such as dipole moments and partial charges.

In the current implementation of PiNN, the primary focus is on predicting atomic properties. In the near future, we aim to include predictions of electronic properties such as polarization as well as the coupling to the external field in periodic systems. These extensions will be quite important for modeling electrochemical systems with finite field MD simulations.[80,81] Moreover, we are interested in incorporating active learning and on-the-fly potential generation.

Last but not least, PiNN will take advantage of the evolving TensorFlow ecosystem, which allows to access novel optimization methods implemented in its peripheral packages such as K-FAC (Kronecker-factored approximate curvature)[82] for fast training of ANNs.

ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available free of charge at https://pubs.acs.org/doi/10.1021/acs.jcim.9b00994.

Details of liquid water dataset augmentation and AIMD simulations of liquid water (PDF)

AUTHOR INFORMATION

Corresponding Authors

Matti Hellström − *Software for Chemistry and Materials B.V., 1081HV Amsterdam, The Netherlands;* orcid.org/0000-0003-3053-5658; Email: hellstrom@scm.com

Chao Zhang − *Department of Chemistry-Ångström Laboratory, Uppsala University, 75121 Uppsala, Sweden;* orcid.org/0000-0002-7167-0840; Email: chao.zhang@kemi.uu.se

Authors

Yunqi Shao − *Department of Chemistry-Ångström Laboratory, Uppsala University, 75121 Uppsala, Sweden*

Pavlin D. Mitev − *Department of Chemistry-Ångström Laboratory, Uppsala University, 75121 Uppsala, Sweden;* orcid.org/0000-0002-4315-8073

Lisanne Knijff − *Department of Chemistry-Ångström Laboratory, Uppsala University, 75121 Uppsala, Sweden*

Complete contact information is available at: https://pubs.acs.org/10.1021/acs.jcim.9b00994

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

C.Z. is grateful to Uppsala University for a start-up grant, to the Swedish Research Council for a starting grant (no. 2019-05012), and to the Swedish National Strategic e-Science program eSSENCE for funding. M.H. received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 798129. Supports from NVIDIA Corporation GPU grant program and Google Cloud Platform (GCP) research credits award are also gratefully acknowledged. Part of the simulations were performed on the resources provided by the Swedish National Infrastructure for Computing (SNIC) at UPPMAX and PDC.

REFERENCES

(1) Dirac, P. A. M. Quantum mechanics of many-electron systems. *Proc. R. Soc. London, Ser. A* **1929**, *123*, 714−733.

(2) Bartók, A. P.; De, S.; Poelking, C.; Bernstein, N.; Kermode, J. R.; Csányi, G.; Ceriotti, M. Machine learning unifies the modeling of materials and molecules. *Sci. Adv.* **2017**, *3*, No. e1701816.

(3) Aspuru-Guzik, A.; Lindh, R.; Reiher, M. The Matter Simulation (R)evolution. *ACS Cent. Sci.* **2018**, *4*, 144−152.

(4) Butler, K. T.; Davies, D. W.; Cartwright, H.; Isayev, O.; Walsh, A. Machine learning for molecular and materials science. *Nature* **2018**, *559*, 547.

(5) Behler, J.; Parrinello, M. Generalized Neural-Network Representation of High-Dimensional Potential-Energy Surfaces. *Phys. Rev. Lett.* **2007**, *98*, 146401.

(6) Schütt, K. T.; Arbabzadah, F.; Chmiela, S.; Müller, K.-R.; Tkatchenko, A. Quantum-chemical insights from deep tensor neural networks. *Nat. Commun.* **2017**, *8*, 13890.

(7) Gassner, H.; Probst, M.; Lauenstein, A.; Hermansson, K. Representation of Intermolecular Potential Functions by Neural Networks. *J. Phys. Chem. A* **1998**, *102*, 4596−4605.

(8) Behler, J. Atom-centered symmetry functions for constructing high-dimensional neural network potentials. *J. Chem. Phys.* **2011**, *134*, 074106.

(9) Behler, J. Constructing High-Dimensional Neural Network Potentials: A Tutorial *Review*. *Int. J. Quantum Chem.* **2015**, *115*, 1032−1050.

(10) Smith, J. S.; Isayev, O.; Roitberg, A. E. ANI-1: an extensible neural network potential with DFT accuracy at force field computational cost. *Chem. Sci.* **2017**, *8*, 3192−3203.

(11) Gastegger, M.; Behler, J.; Marquetand, P. Machine learning molecular dynamics for the simulation of infrared spectra. *Chem. Sci.* **2017**, *8*, 6924−6935.
(12) Yao, K.; Herr, J. E.; Toth, D. W.; Mckintyre, R.; Parkhill, J. The TensorMol-0.1 model chemistry: a neural network augmented with long-range physics. *Chem. Sci.* **2018**, *9*, 2261−2269.
(13) Schran, C.; Uhl, F.; Behler, J.; Marx, D. High-dimensional neural network potentials for solvation: The case of protonated water clusters in helium. *J. Chem. Phys.* **2018**, *148*, 102310.
(14) Khaliullin, R. Z.; Eshet, H.; Kühne, T. D.; Behler, J.; Parrinello, M. Nucleation mechanism for the direct graphite-to-diamond phase transition. *Nat. Mater.* **2011**, *10*, 693−697.
(15) Artrith, N.; Kolpak, A. M. Grand canonical molecular dynamics simulations of Cu-Au nanoalloys in thermal equilibrium using reactive ANN potentials. *Comput. Mater. Sci.* **2015**, *110*, 20.
(16) Morawietz, T.; Singraber, A.; Dellago, C.; Behler, J. How van der Waals interactions determine the unique properties of water. *Proc. Natl. Acad. Sci. U. S. A.* **2016**, *113*, 8368−8373.
(17) Hellström, M.; Quaranta, V.; Behler, J. One-dimensional vs. two-dimensional proton transport processes at solid−liquid zinc-oxide−water interfaces. *Chem. Sci.* **2019**, *10*, 1232−1243.
(18) Krizhevsky, A.; Sutskever, I.; Hinton, G. E. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*; Curran Associates, Inc., 2012; pp 1097−1105.
(19) Duvenaud, D. K.; Maclaurin, D.; Iparraguirre, J.; Bombarell, R.; Hirzel, T.; Aspuru-Guzik, A.; Adams, R. P. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in Neural Information Processing Systems 28*; Curran Associates, Inc., 2015; pp 2224−2232.
(20) Kearnes, S.; McCloskey, K.; Berndl, M.; Pande, V.; Riley, P. Molecular graph convolutions: moving beyond fingerprints. *J. Comput.-Aided Mol. Des.* **2016**, *30*, 595−608.
(21) Schütt, K. T.; Sauceda, H. E.; Kindermans, P.-J.; Tkatchenko, A.; Müller, K.-R. SchNet—A deep learning architecture for molecules and materials. *J. Chem. Phys.* **2018**, *148*, 241722.
(22) Xie, T.; Grossman, J. C. Crystal Graph Convolutional Neural Networks for an Accurate and Interpretable Prediction of Material Properties. *Phys. Rev. Lett.* **2018**, *120*, 145301.
(23) Lubbers, N.; Smith, J. S.; Barros, K. Hierarchical modeling of molecular energies using a deep neural network. *J. Chem. Phys.* **2018**, *148*, 241715.
(24) Unke, O. T.; Meuwly, M. PhysNet: A Neural Network for Predicting Energies, Forces, Dipole Moments, and Partial Charges. *J. Chem. Theory Comput.* **2019**, *15*, 3678−3693.
(25) Chen, C.; Ye, W.; Zuo, Y.; Zheng, C.; Ong, S. P. Graph Networks as a Universal Machine Learning Framework for Molecules and Crystals. *Chem. Mater.* **2019**, *31*, 3564−3572.
(26) Zubatyuk, R.; Smith, J. S.; Leszczynski, J.; Isayev, O. Accurate and transferable multitask prediction of chemical properties with an atoms-in-molecules neural network. *Sci. Adv.* **2019**, *5*, eaav6490.
(27) Chen, X.; Jørgensen, M. S.; Li, J.; Hammer, B. Atomic Energies from a Convolutional Neural Network. *J. Chem. Theory Comput.* **2018**, *14*, 3933−3942.
(28) Schütt, K. T.; Gastegger, M.; Tkatchenko, A.; Müller, K.-R. *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*; Samek, W.; Montavon, G.; Vedaldi, A.; Hansen, L. K.; Müller, K.-R., Eds.; Springer International Publishing: Cham, 2019, pp 311−330.
(29) Khorshidi, A.; Peterson, A. A. Amp: A modular approach to machine learning in atomistic simulations. *Comput. Phys. Commun.* **2016**, *207*, 310−324.
(30) Artrith, N.; Urban, A. An implementation of artificial neural-network potentials for atomistic materials simulations: Performance for TiO2. *Comput. Mater. Sci.* **2016**, *114*, 135−150.
(31) Wang, H.; Zhang, L.; Han, J.; Weinan, E. DeePMD-kit: A deep learning package for many-body potential energy representation and molecular dynamics. *Comput. Phys. Commun.* **2018**, *228*, 178−184.
(32) Singraber, A.; Behler, J.; Dellago, C. Library-Based LAMMPS Implementation of High-Dimensional Neural Network Potentials. *J. Chem. Theory Comput.* **2019**, *15*, 1827−1840.
(33) Schütt, K. T.; Kessel, P.; Gastegger, M.; Nicoli, K. A.; Tkatchenko, A.; Müller, K.-R. SchNetPack: A Deep Learning Toolbox For Atomistic Systems. *J. Chem. Theory Comput.* **2019**, *15*, 448−455.
(34) Huang, B.; Von Lilienfeld, O. A. Communication: Understanding molecular representations in machine learning: The role of uniqueness and target similarity. *J. Chem. Phys.* **2016**, *145*, 161102.
(35) Gastegger, M.; Schwiedrzik, L.; Bittermann, M.; Berzsenyi, F.; Marquetand, P. wACSF—Weighted atom-centered symmetry functions as descriptors in machine learning potentials. *J. Chem. Phys.* **2018**, *148*, 241709.
(36) Faber, F. A.; Christensen, A. S.; Huang, B.; Von Lilienfeld, O. A. Alchemical and structural distribution based representation for universal quantum machine learning. *J. Chem. Phys.* **2018**, *148*, 241717.
(37) Zhang, L.; Han, J.; Wang, H.; Car, R.; Weinan, E. Deep Potential Molecular Dynamics: A Scalable Model with the Accuracy of Quantum Mechanics. *Phys. Rev. Lett.* **2018**, *120*, 143001.
(38) Willatt, M. J.; Musil, F.; Ceriotti, M. Atom-density representations for machine learning. *J. Chem. Phys.* **2019**, *150*, 154110.
(39) Bartók, A. P.; Kondor, R.; Cányi, G. On representing chemical environments. *Phys. Rev. B* **2013**, *87*, 184115.
(40) Drautz, R. Atomic cluster expansion for accurate and transferable interatomic potentials. *Phys. Rev. B* **2019**, *99*, No. 014104.
(41) Glielmo, A.; Zeni, C.; De Vita, A. Efficient nonparametric *n*-body force fields from machine learning. *Phys. Rev. B* **2018**, *97*, 184307.
(42) Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; Dahl, G. E. Neural Message Passing for Quantum Chemistry. In *Proceedings of the 34th International Conference on Machine Learning*; *70*, 2017, pp 1263−1272.
(43) Xu, K.; Hu, W.; Leskovec, J.; Jegelka, S. How Powerful are Graph Neural Networks? In *International Conference on Learning Representations*; Cornell University, 2019.
(44) Abadi, M.; et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. 2016, *arXiv:1603.04467 [cs.DC]*. arXiv.org e-Print archive. https://arxiv.org/abs/1603.04467.
(45) *PiNN documentation*. https://teoroo-pinn.readthedocs.io/en/latest (accessed 2019-10-08).
(46) Ramakrishnan, R.; Dral, P. O.; Rupp, M.; Von Lilienfeld, O. A. Quantum chemistry structures and properties of 134 kilo molecules. *Sci. Data* **2014**, *1*, 140022.
(47) Smith, J. S.; Isayev, O.; Roitberg, A. E. ANI-1, A data set of 20 million calculated off-equilibrium conformations for organic molecules. *Sci. Data* **2017**, *4*, 170193.
(48) van der Walt, S.; Colbert, S. C.; Varoquaux, G. The NumPy Array: A Structure for Efficient Numerical Computation. *Comput. Sci. Eng.* **2011**, *13*, 22−30.
(49) Larsen, A. H.; et al. The atomic simulation environment—a Python library for working with atoms. *J. Phys.: Condens. Matter* **2017**, *29*, 273002.
(50) Behler, J. First Principles Neural Network Potentials for Reactive Simulations of Large Molecular and Condensed Systems. *Angew. Chem., Int. Ed.* **2017**, *56*, 12828−12840.
(51) Hutter, J.; Iannuzzi, M.; Schiffmann, F.; VandeVondele, J. CP2K: atomistic simulations of condensed matter systems. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **2014**, *4*, 15−25.
(52) Rüger, R.; Franchini, M.; Trnka, T.; Yakovlev, A.; van Lenthe, E.; Philipsen, P.; van Vuren, T.; Klumpers, B.; Soini, T. SCM, Theoretical Chemistry. In *AMS 2019*; Vrije Universiteit: Amsterdam, The Netherlands, http://www.scm.com.
(53) Allen, M. P.; Tildesley, D. J. *Computer Simulation of Liquids*; Second Edition; Oxford University Press, 2017.
(54) Kapil, V.; et al. i-PI 2.0: A universal force engine for advanced molecular simulations. *Comput. Phys. Commun.* **2019**, *236*, 214−223.

https://dx.doi.org/10.1021/acs.jcim.9b00994
J. Chem. Inf. Model. 2020, 60, 1184−1193

(55) Thompson, A. P.; Plimpton, S. J.; Mattson, W. General formulation of pressure and stress tensor for arbitrary manybody interaction potentials under periodic boundary conditions. *J. Chem. Phys.* **2009**, *131*, 154107.
(56) Martin, R. M. *Electronic Structure: Basic Theory and Practical Methods*; Cambridge University Press: Cambridge, 2013.
(57) Zeiler, M. D.; Fergus, R. Visualizing and Understanding Convolutional Networks. In *Computer Vision − ECCV 2014*; Springer: Cham, 2014, pp 818−833, DOI: 10.1007/978-3-319-10590-1_53.
(58) Gastegger, M.; Marquetand, P. Molecular Dynamics with Neural-Network Potentials. 2018, *arXiv:1812.07676 [physics, stat]*. arXiv.org e-Print archive. https://arxiv.org/abs/1812.07676.
(59) Kingma, D. P.; Ba, J. Adam: A Method for Stochastic Optimization. 2017, *arXiv:1412.6980 [cs]*. arXiv.org e-Print archive. https://arxiv.org/abs/1412.6980.
(60) Pascanu, R.; Mikolov, T.; Bengio, Y. On the Difficulty of Training Recurrent Neural Networks. *Proceedings of the 30th International Conference on International Conference on Machine Learning* **2013**, *28*, 1310−1318.
(61) Becke, A. D. Density-functional thermochemistry. III. The role of exact exchange. *J. Chem. Phys.* **1993**, *98*, 5648−5652.
(62) Lee, C.; Yang, W.; Parr, R. G. Development of the Colle-Salvetti correlation-energy formula into a functional of the electron density. *Phys. Rev. B* **1988**, *37*, 785−789.
(63) Stephens, P. J.; Devlin, F. J.; Chabalowski, C. F.; Frisch, M. J. Ab Initio Calculation of Vibrational Absorption and Circular Dichroism Spectra Using Density Functional Force Fields. *J. Phys. Chem.* **1994**, *98*, 11623−11627.
(64) Marenich, A. V.; Jerome, S. V.; Cramer, C. J.; Truhlar, D. G. Charge Model 5: An Extension of Hirshfeld Population Analysis for the Accurate Description of Molecular Interactions in Gaseous and Condensed Phases. *J. Chem. Theory Comput.* **2012**, *8*, 527−541.
(65) Frisch, M. J.; et al. *Gaussian 09* Revision D.01. Gaussian Inc.: Wallingford CT, 2009.
(66) Sifain, A. E.; Lubbers, N.; Nebgen, B. T.; Smith, J. S.; Lokhov, A. Y.; Isayev, O.; Roitberg, A. E.; Barros, K.; Tretiak, S. Discovering a Transferable Charge Assignment Model Using Machine Learning. *J. Phys. Chem. Lett.* **2018**, *9*, 4495−4501.
(67) Jain, A.; Ong, S. P.; Hautier, G.; Chen, W.; Richards, W. D.; Dacek, S.; Cholia, S.; Gunter, D.; Skinner, D.; Ceder, G.; Persson, K. A. Commentary: The Materials Project: A materials genome approach to accelerating materials innovation. *APL Mater.* **2013**, *1*, No. 011002.
(68) Kirklin, S.; Saal, J. E.; Meredig, B.; Thompson, A.; Doak, J. W.; Aykol, M.; Rühl, S.; Wolverton, C. The Open Quantum Materials Database (OQMD): assessing the accuracy of DFT formation energies. *npj Comput. Mater.* **2015**, *1*, 15010.
(69) Castelli, I. E.; Olsen, T.; Datta, S.; Landis, D. D.; Dahl, S.; Thygesen, K. S.; Jacobsen, K. W. Computational screening of perovskite metal oxides for optimal solar light capture. *Energy Environ. Sci.* **2012**, *5*, 5814−5819.
(70) Xie, T.; Grossman, J. C. Hierarchical visualization of materials space with graph convolutional neural networks. *J. Chem. Phys.* **2018**, *149*, 174111.
(71) Morawietz, T.; Behler, J. *HDNNP training data set for H2O*. 2019; https://zenodo.org/record/2634098.
(72) Becke, A. D. Density-functional exchange-energy approximation with correct asymptotic behavior. *Phys. Rev. A* **1988**, *38*, 3098−3100.
(73) Berendsen, H. J. C.; Postma, J. P. M.; van Gunsteren, W. F.; DiNola, A.; Haak, J. R. Molecular dynamics with coupling to an external bath. *J. Chem. Phys.* **1984**, *81*, 3684−3690.
(74) Hellström, M.; Behler, J. Concentration-Dependent Proton Transfer Mechanisms in Aqueous NaOH Solutions: From Acceptor-Driven to Donor-Driven and Back. *J. Phys. Chem. Lett.* **2016**, *7*, 3302−3306.
(75) Hammer, B.; Hansen, L. B.; Nørskov, J. K. Improved adsorption energetics within density-functional theory using revised Perdew-Burke-Ernzerhof functionals. *Phys. Rev. B* **1999**, *59*, 7413−7421.
(76) Grimme, S.; Antony, J.; Ehrlich, S.; Krieg, H. A consistent and accurate ab initio parametrization of density functional dispersion correction (DFT-D) for the 94 elements H-Pu. *J. Chem. Phys.* **2010**, *132*, 154104.
(77) Hellström, M.; Behler, J. Structure of aqueous NaOH solutions: insights from neural-network-based molecular dynamics simulations. *Phys. Chem. Chem. Phys.* **2017**, *19*, 82−96.
(78) Hellström, M.; Ceriotti, M.; Behler, J. Nuclear Quantum Effects in Sodium Hydroxide Solutions from Neural Network Molecular Dynamics Simulations. *J. Phys. Chem. B* **2018**, *122*, 10158−10171.
(79) Tuckerman, M. E.; Marx, D.; Parrinello, M. The nature and transport mechanism of hydrated hydroxide ions in aqueous solution. *Nature* **2002**, *417*, 925−929.
(80) Zhang, C.; Hutter, J.; Sprik, M. Coupling of Surface Chemistry and Electric Double Layer at TiO_2 Electrochemical Interfaces. *J. Phys. Chem. Lett.* **2019**, *10*, 3871−3876.
(81) Zhang, C.; Hutter, J.; Sprik, M. Computing the Kirkwood g-Factor by Combining Constant Maxwell Electric Field and Electric Displacement Simulations: Application to the Dielectric Constant of Liquid Water. *J. Phys. Chem. Lett.* **2016**, *7*, 2696−2701.
(82) Martens, J.; Grosse, R. Optimizing Neural Networks with Kronecker-Factored Approximate Curvature. *Proceedings of the 32nd International Conference on International Conference on Machine Learning* **2015**, *37*, 2408−2417.

https://dx.doi.org/10.1021/acs.jcim.9b00994
J. Chem. Inf. Model. 2020, 60, 1184−1193