Digital Discovery

ROYAL SOCIETY OF CHEMISTRY

PAPER

View Article Online
View Journal | View Issue

Check for updates

Cite this: *Digital Discovery*, 2025, **4**, 2502

Received 5th March 2025
Accepted 16th July 2025

DOI: 10.1039/d5dd00085h

rsc.li/digitaldiscovery

Teacher-student training improves the accuracy and efficiency of machine learning interatomic potentials

Sakib Matin,[*a] Alice E. A. Allen,[abc] Emily Shinkle,[d] Aleksandra Pachalieva,[e] Galen T. Craven,[a] Benjamin Nebgen,[a] Justin S. Smith,[f] Richard Messerly,[a] Ying Wai Li,[d] Sergei Tretiak,[abg] Kipton Barros[ab] and Nicholas Lubbers[d]

Machine learning interatomic potentials (MLIPs) are revolutionizing the field of molecular dynamics (MD) simulations. Recent MLIPs have tended towards more complex architectures trained on larger datasets. The resulting increase in computational and memory costs may prohibit the application of these MLIPs to perform large-scale MD simulations. Herein, we present a teacher-student training framework in which the latent knowledge from the teacher (atomic energies) is used to augment the students' training. We show that the light-weight student MLIPs have faster MD speeds at a fraction of the memory footprint compared to the teacher models. Remarkably, the student models can even surpass the accuracy of the teachers, even though both are trained on the same quantum chemistry dataset. Our work highlights a practical method for MLIPs to reduce the resources required for large-scale MD simulations.

1 Introduction

Molecular Dynamics (MD)[1] is ubiquitous in chemistry,[2] materials science,[3] and drug discovery[4] as well as other fields. Accurate chemical and thermodynamic properties derived from MD rely on accurate interatomic potentials, which parameterize the many-body interactions present between atoms.[1,5] Simulation scales may vary greatly depending on the questions of interest and available resources, and there is a persistent need for greater model efficiency. Traditional classical potentials are very fast, making it possible to perform large-scale simulations of billions of atom,[6] or to perform hundreds of millions of integration time-steps for small systems. Recently, there has been a great demand for interatomic potentials that are more accurate *via* machine learning models that are trained to reference quantum mechanical force calculations. Here, the goals of efficiency and accuracy can be in conflict. Our paper is concerned with techniques for improving the efficiency of the interatomic potential without sacrificing the accuracy.

The gold-standard for computational chemistry is *ab initio* molecular dynamics, which uses quantum chemistry (QC) methods for accurately calculating interatomic forces from first principles.[7,8] However, the often prohibitive computational cost of QC methods and its rapid growth with system size limit the size of the systems that can be simulated. Machine learning interatomic potentials (MLIPs)[9–13] can be trained on QC datasets to map from an atomic configuration to energy and forces. MLIPs can achieve the chemical accuracy (error <1 kcal mol^{-1}) of QC simulations[14] at drastically reduced computational costs. Most MLIPs achieve linear scaling with system size by utilizing the approximation that the total predicted energy can be decomposed into a sum of spatially local atom-wise or pair-wise contributions.[11,12,15,16] MLIPs have seen explosive growth and have been successfully applied to predicting potential energy surfaces,[17–25] with extensions to a variety of quantities, such as charges[26–30] and more.[31–35] However, MLIPs only mitigate the cost of QC—they do not eliminate it. Generating training data for MLIPs is costly due to the steep scaling of QC methods in system size N, *e.g.*, $\mathcal{O}(N^7)$ at the CCSD(T) level of theory.[7] Therefore, any approach which uses data more efficiently can potentially reduce the overall computational costs of MLIPs.

Furthermore, a recent trend in the field is the significant effort to construct foundation model MLIPs[36–38] with broad chemical coverage,[39] inspired by the success of large pre-trained models in natural language processing.[40] Foundation MLIPs,[41] which may be parameterized by up to 10^9 fitting parameters,[42]

[a] *Los Alamos National Laboratory, Theoretical Division, Los Alamos, New Mexico, USA. E-mail: sakibmatin@gmail.com*

[b] *Los Alamos National Laboratory, Center for Nonlinear Studies, Los Alamos, New Mexico 87546, USA*

[c] *Max Planck Institute for Polymer Research, Ackermannweg 10, 55128 Mainz, Germany*

[d] *Los Alamos National Laboratory, Computer, Computational, and Statistical Sciences Division, Los Alamos, New Mexico, USA*

[e] *Los Alamos National Laboratory, Earth and Environmental Sciences Division, Los Alamos, New Mexico, USA*

[f] *Nvidia Corporation, Santa Clara, California, 9505, USA*

[g] *Los Alamos National Laboratory, Center for Integrated Nanotechnologies, Los Alamos, New Mexico 87546, USA*

have high computational and memory requirements.[43,44] These increasing computational and memory costs compared to classical force fields[1] can limit the usability of MLIPs for large-scale MD simulations.[13,43–45]

In this manuscript, we introduce a teacher-student training method for MLIP with a central purpose of building the MLIPs with faster inference and lower memory requirements. The teacher-student training is a class of methods[46–48] to reduce the inference costs of different ML models and more effectively utilize existing datasets. An initial teacher model is trained and then used to augment the training of a student model that may have faster inference,[43,44,48] smaller memory requirements,[49] or better generalization capacity.[47] Crucially, this knowledge distillation procedure does not require additional first-principles training data. Instead, auxiliary predictions of the teacher model are used to augment the data used for training of the student model. The innovation in this work is to use the teacher's local atomic energy predictions as the auxiliary training data. Although these local atomic energies have traditionally been considered a latent feature of the MLIP more,[11] the present work highlights that they carry important information. Note that the latent atomic energies provided to the student are far greater in number than the single global QC energy. Thus, the student is trained using a significantly larger number of constraints than the teacher. Because the student model will typically have fewer trainable weights than the teacher model, this approach can yield significant gains for inference speed and memory requirements. We also find that the student model can achieve a higher accuracy than the teacher. An overview of our workflow is presented in Fig. 1.

2 Methods & background

We apply the teacher-student training to an MLIP architecture, namely the Hierarchically Interacting Particle Neural Network (HIPNN).[14,22] HIPNN is a message passing graph neural network used to model atomistic systems,[22,28] and recent variants incorporate equivariant tensor sensitivity for higher accuracy.[14] The architecture of HIPNN is briefly reviewed in Section 2.1. In Section 2.2, we introduce the teacher-student paradigm. Section 2.3 summarizes the training procedure. Section 2.4 details the datasets studied. Section 3 contains a systematic exploration of the teacher-student training for the HIPNN model.

2.1 Architecture

HIPNN[22] is a message-passing graph neural network[50] that can map atomic configurations to various chemical properties such as energy,[22] forces,[14] dipoles,[28] *etc.* Similar to many MLIP architectures, the interaction layers of HIPNN allow for mixing of atomic environments between neighbors *via* message passing to construct the learnable features.[22] The local atomic environment of each atom is initially featurized using atomic

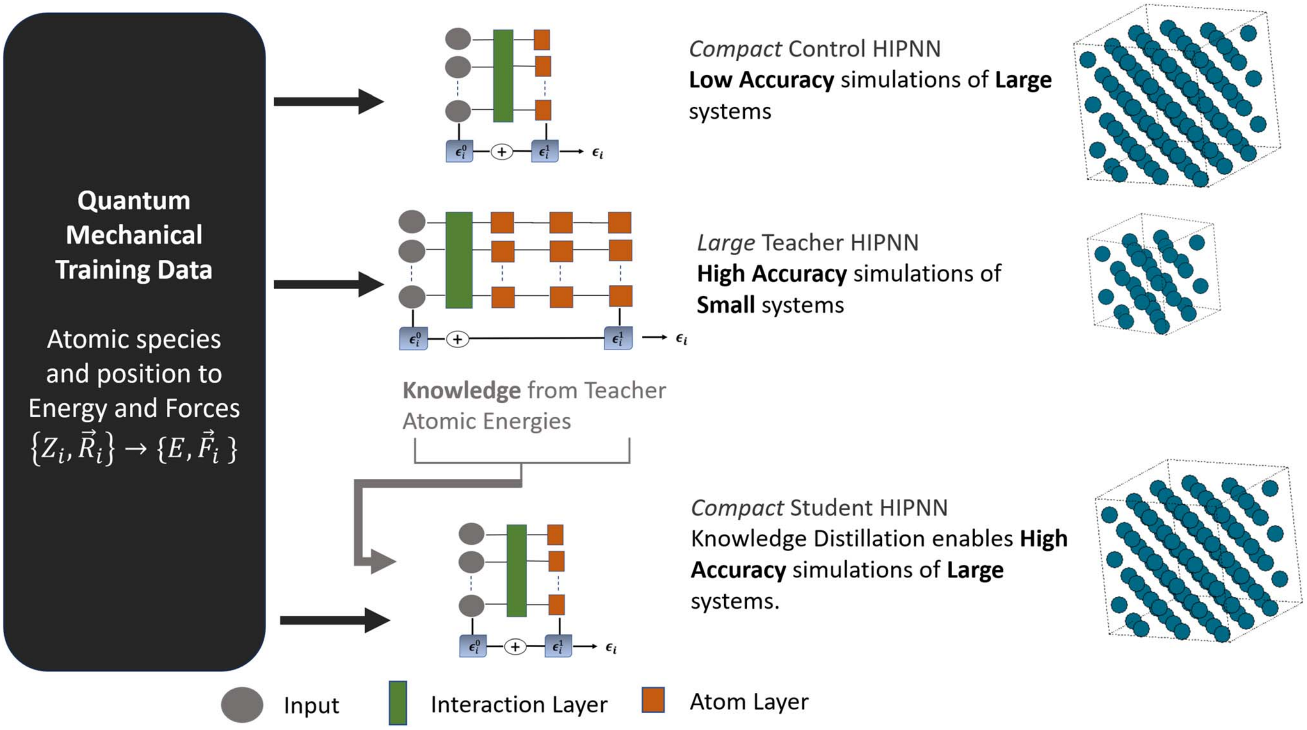
Quantum Mechanical Training Data
Atomic species and position to Energy and Forces
$\{Z_i, \vec{R}_i\} \rightarrow \{E, \vec{F}_i\}$
ϵ_i^0
ϵ_i^1
ϵ_i
Compact Control HIPNN
Low Accuracy simulations of **Large** systems
Large Teacher HIPNN
High Accuracy simulations of **Small** systems
Knowledge from Teacher Atomic Energies
Compact Student HIPNN
Knowledge Distillation enables **High Accuracy** simulations of **Large** systems.
Input
Interaction Layer
Atom Layer

Fig. 1 Teacher-student training for the Hierarchically Interacting Particle Neural Network (HIPNN) machine learning interatomic potential. A HIPNN model consists of an input node, a message passing interaction layer and feed-forward regression layers called atom layers. The teacher HIPNN is trained on the quantum mechanical energy and force data and generates latent knowledge in the form of atomic energies. This augments the student model training to improve the accuracy. The student HIPNN that has fewer trainable weights contributes to faster inference and lower memory requirements. The control HIPNNs have the same number of trainable parameters as the student but are trained only on ground truth data. We show that the student models are more accurate than the control models.

number (in HIPNN, as a one-hot encoding, although also common is a random embedding) and passed through several layers along with pair-wise displacement vectors between the neighbors with local cut-off to predict the atomic energy ε_i for each atom i. Automatic differentiation is used to calculate the forces on each atom from the total molecular energy.

MLIP architectures almost universally infer a local decomposition in their predictions of extensive quantities. In message passing neural network parlance, the readout function is a linear summation over node states. Specifically, in HIPNN, the energy $E[r_1, r_2, \ldots, r_N]$ of a configuration, where N is the number of atoms, is decomposed into a sum of local contributions:

$$E \approx \hat{E} = \sum_{i}^{N} \varepsilon_i. \tag{1}$$

The energy contributions are formed by a linear combination of features (also called embeddings) learned by the neural network. The teacher-student procedure we investigate here depends on the existence of some local energy decomposition but is flexible about its details. For example, our procedure could also work on models such as Allegro that decompose energy into contributions of local bonds.[16]

In the original publication,[22] the HIPNN only used scalar pair-wise distances between neighbors, which captures a subset of higher-order many-body information contained in the local chemical environment.[14,41] Subsequently, the HIPNN with tensor sensitivity[14] utilizes higher order tensor products of the displacement vectors between atoms to construct more informative descriptors. The hyper-parameter $l_{\max}$ corresponds to the highest order tensor used in the HIPNN. A weight tying scheme between different order sensitivity functions only leads to a modest increase in the number of trainable parameters, thereby reducing the training and evaluation costs and improving the data efficiency.[14] The $l_{\max} = 0$ HIPNN model utilizes only scalar pair-wise distances and coincides with the model developed in the original publication.[22] In this work, we select $l_{\max} = 1$ to allow for vector sensitivity. Other hyper-parameters for the HIPNN models used in this paper are given in Section 5.

2.2 Teacher-student training

In the teacher-student training method, a pre-trained teacher model (or models) is used to train a student model to improve the speed,[43,44,48] memory requirements,[49] and generalization.[47] Knowledge Distillation (KD)[46] is a well-known form of teacher-student training. In the original KD publication,[46] auxiliary targets are the teacher model's predicted relative class probabilities (before the application of the softmax operation in the output layer). The auxiliary targets generated by the teacher model contain richer information about the structure of the classes in the dataset. The student models are trained on both the ground truth data and the auxiliary targets and perform better than the control models, which have the same architecture as the student model but are trained only on the ground truth data. Later studies have incorporated deeper architecture dependent knowledge.[48,49,52] KD has been successfully applied to many different architectures such as CNN,[53] graph neural networks,[48] transformers[49] and more.[54] Another variant of the teacher-student training is the "Born Again" (BA) method,[47] where the student and teacher models have the same architecture, and the student model can surpass the teacher's accuracy. Multiple teachers can be used to train a single student model to improve the performance.[47,55–57] Previous studies focus on classification tasks on images[46] and text[49] and limit applications to regression tasks on graph structured data.[58]

Recently, teacher-student methods have been explored in chemistry for accelerating molecular dynamics,[43,44] physics-constrained data augmentation,[59] and material property prediction.[60] In a related work in ref. 61, the teacher MLIP (trained on the QC ground truth) is used to generate synthetic data by running MD under different conditions. The student model is initially pre-trained on the synthetic data and then fine-tuned on the QC ground truth.

In the first step of our teacher-student method, we train a complex "teacher" MLIP on a QC dataset, which consists of configuration energy and forces on each atom. This teacher MLIP can generate auxiliary targets, namely per atom energies. The atomic energy may provide more fine-grained information than the aggregate configuration energy.[43,61,62] Then, we train the student MLIP on the original QC data and auxiliary targets generated by the teacher MLIP. This economical approach does not require any expensive QC calculations beyond the original dataset needed to train the teacher model, nor exhaustive hyper-parameter tuning.

2.3 Training procedure and loss function

We train the teacher model $\mathscr{T}$ on the QC dataset $\mathfrak{D}$, which contains a set of atomic positions and species for each configuration and the corresponding energy and forces per atom, $\mathfrak{D}: \{\mathbf{R}_i, Z_i\} \rightarrow \{E, \mathbf{F}_i\}$..

The NN model is trained in a standard manner using stochastic gradient descent on the loss function

$$\mathscr{L}_{\text{teacher}} = w_E \mathscr{L}_{\text{err}}\left(\hat{E}, E\right) + w_F \mathscr{L}_{\text{err}}\left(\hat{F}, F\right) + w_{\mathrm{L}_2}\mathscr{L}_{\mathrm{L}_2} + w_{\mathrm{R}}\mathscr{L}_{\mathrm{R}}. \tag{2}$$

The $\mathscr{L}_{\mathrm{L}_2}$ loss term is a regularization of the model weights, which is commonly added to loss functions to reduce overfitting. The $\mathscr{L}_{\mathrm{R}}$ term, specific to HIPNN, also enhances model stability. The error loss $\mathscr{L}_{\text{err}}$ could be any combination of common metrics but in our case is an equal weighting between root-mean-squared error (RMSE) and mean-absolute error (MAE) losses:

$$\mathscr{L}_{\text{err}}\left(\hat{V}, V\right) = \mathrm{RMSE}\left(\hat{V}, V\right) + \mathrm{MAE}\left(\hat{V}, V\right), \tag{3}$$

where $\hat{V}$ is the model prediction and V is the ground truth target. Our loss function uses a sum of RMSE and MAE errors, similar to the original HIPNN model publications.[14,22] The weights of each term in the loss are listed in Appendix A.1.

The teacher model $\mathscr{T}$ maps the local atomic environments to atomic energies ε_i, which are summed over to obtain the energy

$\hat{E}$. Although ε_i is not directly a physical observable, it nonetheless captures important information about how the local geometry affects the final prediction of the model.[43,61,62] Here, we use the teacher MLIP's atomic energies as the knowledge to be transferred to the student MLIPs.

The atomic energy ε_i predictions from $\mathscr{T}$ are used to construct the augmented dataset:

$$\tilde{\mathfrak{D}} : \{\boldsymbol{R}_i, Z_i\} \rightarrow \{E, \varepsilon_i^{\mathscr{T}}, \boldsymbol{F}_i\}. \tag{4}$$

Note that the original dataset $\mathfrak{D}$ has the same set of configurations (inputs) as the augmented data $\tilde{\mathfrak{D}}$.

We train the student models $\mathscr{S}$ on the augmented dataset $\tilde{\mathfrak{D}}$ with the loss function:

$$\mathscr{L}_{\text{student}} = w_E \mathscr{L}_{\text{err}}\left(\hat{E}, E\right) + w_F \mathscr{L}_{\text{err}}\left(\hat{F}, F\right) + w_A \mathscr{L}_{\text{err}}(\varepsilon^{\mathscr{S}}, \varepsilon^{\mathscr{T}}) + w_{L_2}\mathscr{L}_{L_2} + w_R \mathscr{L}_R \tag{5}$$

This captures the notion that the student should learn from the teacher using the loss term $\mathscr{L}_{\text{err}}(\varepsilon^{\mathscr{S}}, \varepsilon^{\mathscr{T}})$, which encourages the partitioning of energy among atomic sites in the student, $\varepsilon^{\mathscr{S}}$, to match that of the teacher, $\varepsilon^{\mathscr{T}}$. We explore student models by varying several architectural parameters of the NN, namely, the layer width n_{feature}, the number of sensitivity function n_{d} used for distance embedding, and the number of atom layers used after interaction layers $n_{\text{atom\_layer}}$. To assess the effectiveness of the method, we also train control models $\mathscr{C}$ that have the same architecture as the students $\mathscr{S}$ but are trained only on the dataset $\mathfrak{D}$ using $\mathscr{L}_{\mathscr{T}}$; everything else is held constant in these models except for the student-teacher loss. The accuracy of the control models serves as a benchmark to compare the efficacy of the teacher-student training framework.

2.4 Data

We apply our teacher-student workflow on the ANI-Al dataset,[23] which consists of condensed-phase aluminum geometries, with energies and forces calculated using Density Functional Theory (DFT). The authors of ref. 23 created the dataset using an automated active learning framework to generate adequate coverage of the configurational space. In this workflow, an ensemble of ANI models[20] were trained to the initial DFT dataset, which consisted of random structures. The main loop of the active learning workflow involves running MD simulations with the ensemble under varying thermodynamic conditions (time-dependent temperatures and density schedules) on boxes of ≈50 to ≈250 atoms. New DFT calculations were performed on configurations where the ensemble disagreement exceeded a predefined threshold. Then, the MLIPs were retrained to the expanded dataset. This loop terminates when long MD simulations (250 ps) could be performed without identifying any new configurations with high ensemble disagreement. Over 50 generations of models, the final dataset comprised about 6000 DFT calculations with the Perdew–Burke–Ernzerhof functional. The dataset is available online.[63] More details are available in the original publication describing the construction of the dataset.[23]

3 Results and discussion

3.1 Pareto dominant student MLIPs

Atomistic simulations require accurate and efficient evaluation of energy and forces. Performing MD at large length and time scales is challenging due to the trade-off between accuracy and

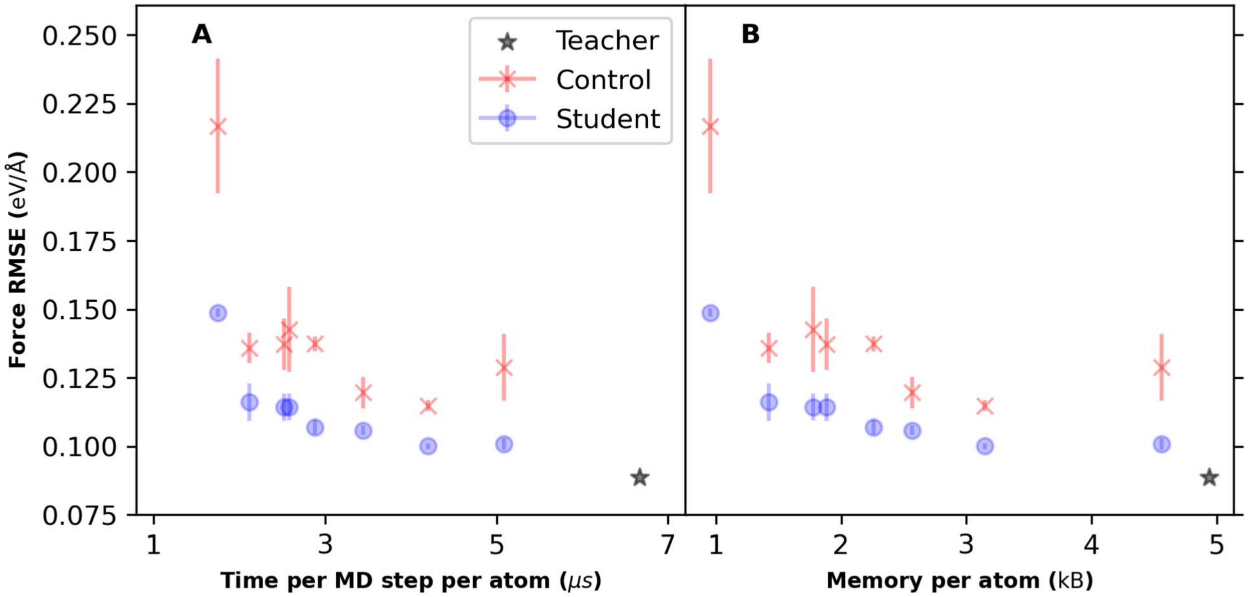

Fig. 2 The student MLIPs are Pareto dominant compared to the control models. The student and control MLIPs are trained on the ANI-Al aluminum dataset.[23] The accuracy metric, force RMSE, is plotted as a function of efficiency metrics, namely, time per MD step per atom in (A) and memory per atom in (B). The origin of the plots corresponds to the Pareto optimum solution. The MD simulations are run using the ASE[51] code on a Nvidia A6000 (48 GB) GPU. The error bar corresponds to the standard deviation of an ensemble of 4 models, which differ by random weight initializations.

efficiency of the interatomic potential used in the simulation. MLIPs provide a path to perform MD with QC accuracy at dramatically reduced costs relative to methods that use explicit QC solutions to calculate the forces and energies that are used to evolve the simulation forward in time. Here, we show that the teacher-student method allows us to improve the accuracy of the smaller MLIPs without increasing the computational costs at inference time.

Characterizing the accuracy of interatomic potentials in absolute terms is challenging.[64] Such comparisons should be made against exact numerical solutions or experimental data, which may not be available. The accuracy of MLIPs can be judged against held-out test data from the QC calculations used to develop the potential. In this section, we use the force root mean squared error (RMSE) as an accuracy measure. We analyze out-of-sample MD-based accuracy metrics in Section 3.2.

It is also difficult to define an unambiguous metric of efficiency for MLIPs. An informative measure is the time-per-MD step per atom, which is affected by hardware (CPU/GPU memory and processing speeds) and software (MD library, algorithm used for neighbor list construction, *etc.*). Therefore, in this article, we use a consistent hardware configuration (a single A6000 GPU with 48 GB of storage), and we use the Atomic Simulation Environment[51] software. We simulate 48 000 atoms at fixed volume and energy (NVE ensemble) with a time step of 1 fs and a simulation time of 1 ps. Additionally, we can characterize the efficiency of MLIPs based on memory requirements. We record the maximum number of atoms that can be simulated on a single A6000 GPU using the ASE software and use it to calculate the average memory-per-atom.

Fig. 2 shows the Pareto plot of accuracy (force RMSE) against the time per MD step per atom and maximum atoms per GPU in panels (A) and (B) respectively. The student models are Pareto dominant with respect to the control MLIPs, *i.e.*, for a given cost (MD speed or memory requirement), the student models have higher accuracy than the control models. Each data point is averaged over four MLIPs initialized with different random seeds. Relative to the teacher, the student MLIPs can simultaneously achieve more than a factor of 2 speed up in MD simulations at less than half the memory requirements while sacrificing less than 20% in force accuracy.

3.2 Accuracy and MLIP capacity

In Fig. 3, we analyze several metrics of accuracy such as force RMSE, energy-per-atom RMSE, and radial distribution function (RDF)[1] error as a function of number of weights of each HIPNN model, which serve as an effective measure of the model capacity. This metric is compelling because it is agnostic to the hardware constraints, such as GPU memory and the choice of MD software. While it is meaningful to compare weights for different variants of an MLIP architecture (HIPNN), care should be taken when making comparisons across different architectures. In Section 3.4, we study how the number of weights correlates with the maximum number of atoms per GPU and MD speeds.

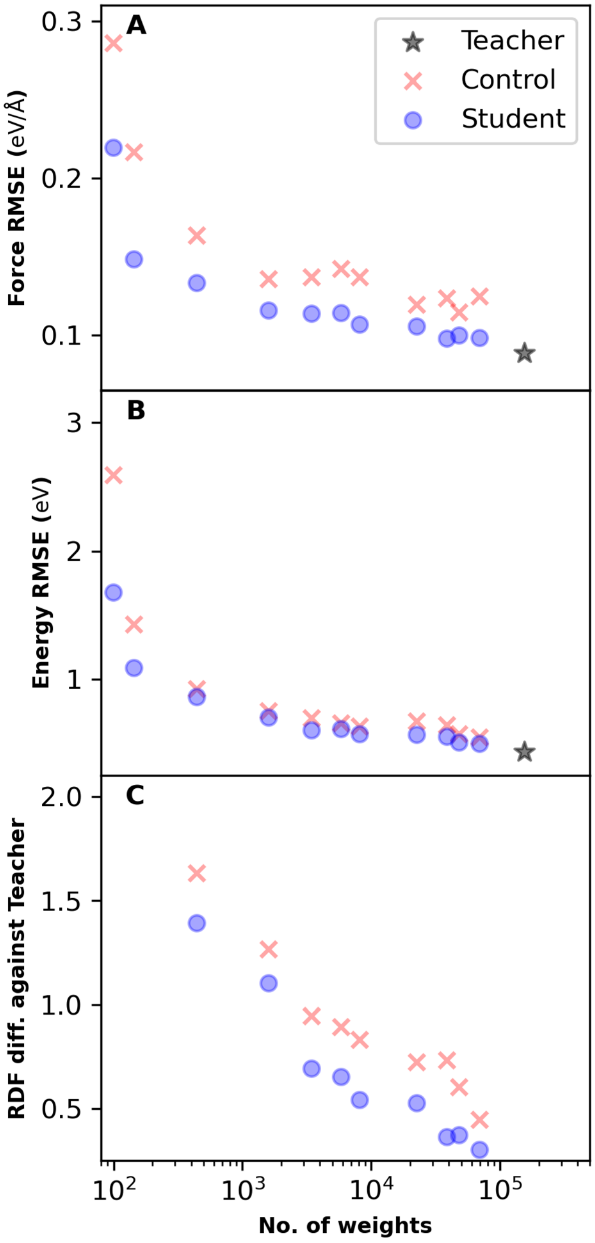

Fig. 3 The student MLIPs are more accurate than the control architectures at the corresponding capacity. The number of trainable weights captures the model capacity. The force RMSE errors and energy per-atom RMSE errors for the training dataset are plotted in panels (A) and (B) respectively. The total absolute error of the radial distribution function (RDF) of the student and control MLIPs compared to the teacher MLIP is shown in panel (C). The error metrics with respect to the training datasets in (A) and (B) show similar trends as the MD-based accuracy metrics in (C).

The force RMSEs in panel (A) of Fig. 3 indicate a meaningful improvement of the student models compared with the control models, while the energy errors in panel (B) are only minimally improved in the student models. We compute the RDFs of liquid aluminum at 1200 K using the LAVA[65] software, which is a wrapper for LAMMPS.[5] The reported RDF errors shown in panel (C) for the student and control models are the total absolute errors with respect to the teacher RDFs. Note that the energy and forces errors in Fig. 3(A) and (B), respectively, are computed with respect to the ground truth training data, whereas the RDF errors in Fig. 3(C) are the differences against the teacher models' MD simulations. Fig. 3 reveals that the energy RMSE is quite similar between the student and control models, except for very small model sizes. We attribute the improvements in the force accuracy to the fact that the atomic energies are local properties predicted for each atom, similar to atomic forces. Furthermore, the force errors strongly correlate with the RDF errors. Note that the RDF simulations constitute a difficult extensibility test because the simulations utilize large periodic boxes that are two orders of magnitude larger than the configurations found in the training data, with each simulation containing 18 634 atoms.

3.3 Learning dynamics

Learning dynamics (also known as optimization dynamics) characterize how the training or out-of-sample validation error evolves as a function of training epochs. We analyze the learning dynamics of student and control MLIPs to understand how the auxiliary targets affect the training. We see in Fig. 4 that the student models' learning dynamics outperform those of the control models. Loss curves of four different student and control models all share the same architecture. Here, we use four different seed values for the random initialization of the sets of four control and four students, respectively. By employing the same seed for a pair of student and control MLIPs, we ensure that each student MLIP has the same initial weight as its corresponding control MLIP. Note that the learning dynamic curves begin after one epoch of training has been done. The apparent improved initialization of the student models is actually due to the improved learning by the student HIPNN models due to the atomic energy targets from the teacher. We also note that the auxiliary targets have a regularization effect, stabilizing the optimization, because the learning curves of the student models exhibit less variance in the later stages of training.

3.4 Speed and scalability

Performing large-scale MD is necessary to address many important scientific questions. While the atomic time-steps are generally on the order of order 1–10 fs, the physical phenomena of interest may span milliseconds or longer and can require billions of atoms. Large-scale MD on modern super-computing clusters relies on the idea of weak scaling,[5] where the simulation is distributed across many nodes. Due to the high latency of inter-node communications, it is beneficial to fit as many atoms on one node or GPU as possible.

Our work shows that the teacher-student procedure lowers the resource required for large-scale MD at a target accuracy. The teacher-student method improves the accuracy of smaller models. The light-weight student models require fewer floating point operations for force evaluations. As a direct consequence, we can now run large-scale MD at a desired accuracy while using fewer computational resources. Fig. 5 examines how the number of weights in the network affect the computational cost for large-scale MD in terms of both speed and memory in panels (A) and (B), respectively. The MD is performed using ASE in the NVE ensemble for 1 ps with a time step of 1 fs. Furthermore, we see that the smaller student models can fit more atoms on a single GPU (A6000 48 GB in Fig. 5). The ability to fit more atoms on a single GPU can improve the efficiency of large MD simulations because inter-node latency can often dominate the computations.[5]

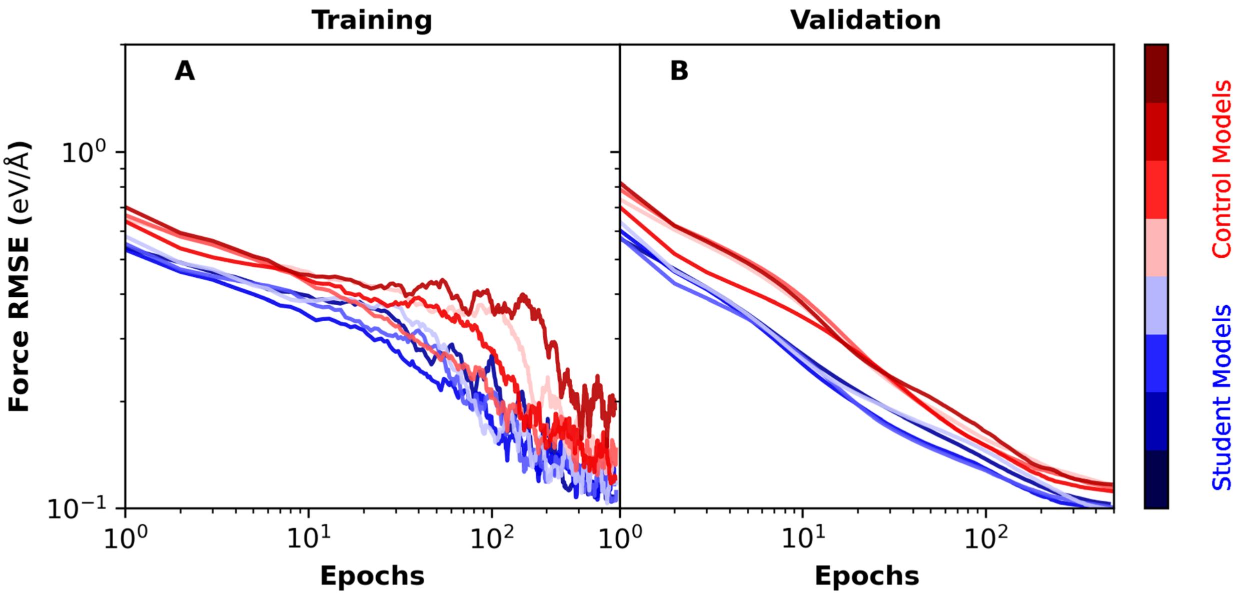

Fig. 4 The student HIPNN MLIP exhibits faster learning dynamics than the control models. Plot of force RMSE for the training and validation data splits are shown as a function of epochs for student and control MLIPs in (A) and (B), respectively.

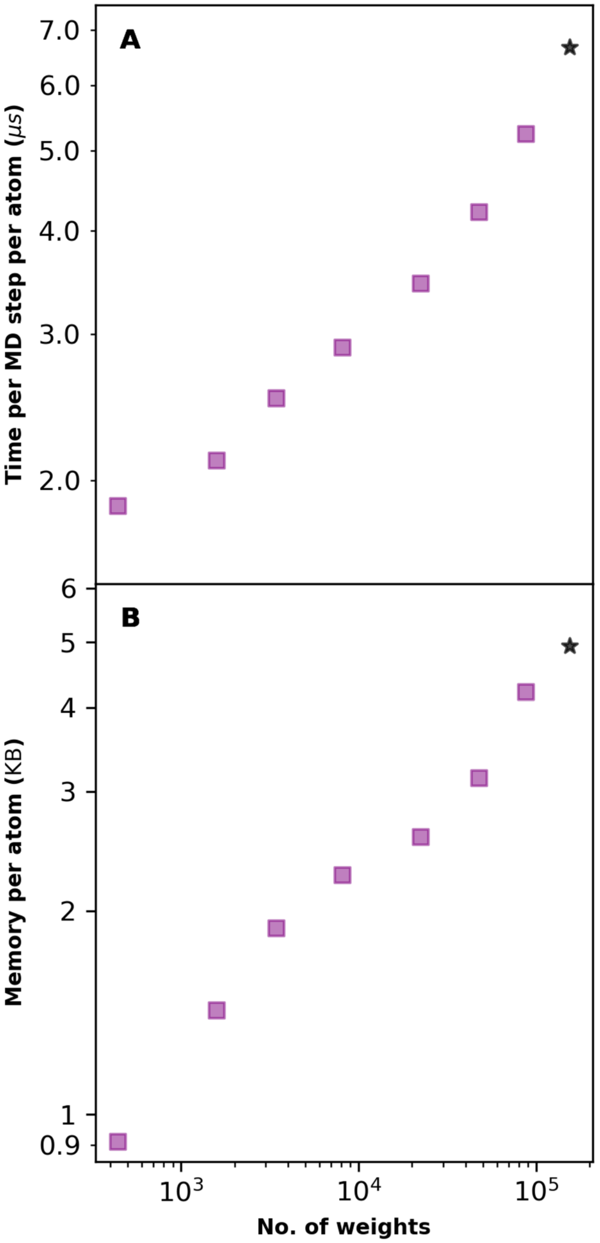

Fig. 5 Inference efficiency as measured by (A) time-per-MD step per atom and (B) memory per atom *versus* the number of trainable weights in HIPNN models. Data are collected by running MD simulations using ASE in the NVE ensemble on a single A6000 GPU (48 GB) for 10^3 steps.

3.5 Student MLIP surpasses the teacher

We apply the "Born Again"[47] teacher-student training to the HIPNN. The 'Born Again' method was introduced in ref. 47, and it is a variant of the teacher-student training where both models have exactly same architecture and number of trainable weights. The central aim of the 'Born Again' method is to train a student model that surpasses the teacher's accuracy. This is in contrast to knowledge distillation, where the aim is to improve the accuracy of smaller student models that have fewer trainable weights than the teacher models. We show that student models trained to the ground truth data and the teacher model's auxiliary outputs and can surpass the teacher by using an loss scheduler.

Table 1 Loss scheduler for student MLIPs for the Born-Again method of teacher-student training in Section 3.5

Epoch	w_A	w_F	w_E
1	200	75	0.0
200	160	63	0.2
250	120	51	0.4
300	80	39	0.6
350	40	27	0.8
400	0	15	1

Initially, we use the static loss function in eqn (2) to find that student MLIPs achieve comparable errors to the teachers. However, then, we utilize a loss scheduler to dynamically update the weights w_A, w_F, and w_E during the training, as summarized in Table 1 in Appendix A.1. The weights in the loss function are chosen to favor the teacher's knowledge in the early stages of training and emphasizing the QM data in the later stages, so that the student can surpass the accuracy on the QM. The values of weight schedule are determined through manual hyper parameter tuning. The final weights of the students' loss function match those of the static weights of the teacher MLIP. The student models' energy RMSE of 0.37 ± 0.02 eV and force RMSE of 0.083 ± 0.003 eV $Å^{-1}$ are lower than the teacher models' errors, energy RMSE of 0.38 eV, and force RMSE of 0.092 eV $Å^{-1}$. These show that the BA teacher-student approach improves the force accuracy by about 10%, which shares the same model architecture and underlying training data.

4 Conclusions and future work

We introduce a teacher-student framework that can be readily applied to many MLIPs that decompose the configurational energy into a sum of local contributions. The teacher-student training framework improves the Pareto set of error-cost trade-offs for MLIPs, yielding models that are computationally cheaper for the same accuracy, or more accurate for the same computational cost, and requires no additional ground-truth data.

In a practical setting, we showed that the student MLIPs are Pareto dominant with respect to the control MLIPs. The student MLIPs, trained to the ground truth and teacher's atomic energies, achieve higher accuracy at the same efficiency (speed and memory requirements) when compared to the control models that were only trained to the ground truth. We use both training errors (energy and force RMSEs) as well as the MD-based metric (RDF errors) to quantify the accuracy of the MLIPs. We find that the force RMSE errors generally correlate with the MD-based metrics, which are costlier to evaluate. MD-based metrics also demonstrate the extensibility of the MLIP to configurations much larger than what is typically included in the training dataset. However, MD-based metrics require a ground truth, such as *ab initio* results or experimental data, which may not always be available. The efficiency of MLIPs is also multifaceted. We use MD speeds and memory-per-atom as effective measures. While MD speed is easy to interpret, it strongly depends on the underlying hardware, MD codes, and the system under study. The memory-per-atom measure is useful to optimize for large MD simulations, where inter-node communications may dominate. While the research to improve the accuracy of MLIPs has focused on the development of more expressive architectures[50] and the generation of larger datasets,[66] we show that the 'Born-Again' inspired teacher-student training can be used to train more accurate models with existing datasets and architectures. Thus, innovations to the training protocol can allow us to extract better models from existing datasets.

We used the atomic energies as the knowledge for the teacher-student training. In the language of message passing graph neural networks, the atomic energy is a node level scalar.[50] In future studies, one can explore node level vectors, such as forces,[56,57] and edge level properties as knowledge for the teacher-student framework to train across different architectures.[43,44] Additionally, network weights for the interactions layers may be utilized as knowledge for the student. One may also explore using an ensemble of teachers for future work, where the students may be able to leverage the uncertainty associated with the teacher MLIPs' knowledge.

Graph neural networks can be used to predict molecular properties beyond just energy and forces such as charges,[29,67] dipoles,[28] and other properties.[11] There is potential to explore cross-modal teacher-student training frameworks to combine teacher models with different specialized tasks to train generalist student MLIPs. This will be an important step towards foundation models for chemistry with broad applicability.

5 Training details

5.1 Hyper-parameters

We use HIPNN models with 1 interaction layer. The teacher models use 4 atom layers (feed-forward layers) with a width of 128. The student (and control) models have between 1 and 4 atom layers with a width between 12 and 128. All models have a maximum tensor sensitivity order set at $l_{\max} = 1$. For the sensitivity functions which parameterize the interaction layer, radial basis functions are used with the soft-min cutoff of 1.5 Å, soft maximum cutoff of 7.0 Å, and hard maximum cutoff of 7.5 Å. The teacher model uses 40 basis functions. For the student (and control) MLIPs, we use between 8 and 40 sensitivity functions. The soft-min cut-off corresponds to the inner cut-off at very short distances. The hard maximum cut-off corresponds to the long distance cut-off. The soft maximum cutoff is set to a value smaller than the hard-dist cutoff to ensure a smooth truncation of the sensitivity functions. Note that we are using the naming conventions for the hyper-parameters in the HIPNN GitHub Repository,[68] which differ slightly from the original HIPNN publication.[22]

We summarize the weights corresponding to the loss function in eqn (2). W_{L_2}=10^{-6} and $W_R = 0.01$ are common to the teacher, student and control models. $W_E = 1$ and $W_F = 10$ are used for the teacher and control MLIPs. Lastly, we use $W_E = 1$, $W_F = 30$ and $W_A = 100$ for the student models. For BA training, we use a loss schedule with weights given in Table 1.

We used the Adam Optimizer, with an initial learning rate of 0.001, which is halved with a patience of 30 epochs. The termination patience is 50 epochs.

Data availability

The HIPNN[14,22] MLIP is implemented in an open-source PyTorch-based software package called hippynn, which is available for download[68] (persistent repository DOI: https://doi.org/10.6084/m9.figshare.29551166.v1). The scripts for workflow can be found in the examples folder. We use an aluminum dataset published in ref. 23, and the dataset is available for download[63] (persistent repository DOI: https://doi.org/10.6084/m9.figshare.29372552.v1).

Author contributions

Conceptualization: SM and NL; methodology: SM, AEAA, ES, AP, GTC, BN, JSS, RM, YWL, ST, KB, and NL; formal analysis: SM; writing – original draft: SM; writing – review and editing: SM, AEAA, ES, AP, GTC, BN, JSS, RM, YWL, ST, KB, and NL; supervision: GTC and NL.

Conflicts of interest

There are no conflicts to declare.

Acknowledgements

This work was supported by the United States Department of Energy (US DOE), Office of Science, Basic Energy Sciences, Chemical Sciences, Geosciences, and Biosciences Division under the Triad National Security, LLC ('Triad') contract grant no. 89233218CNA000001 (FWP: LANLE3F2, LANLE8AN). We acknowledge the Los Alamos National Laboratory (LANL) Directed Research and Development (LDRD) for funding support. This research was performed in part at the Center for Nonlinear Studies (CNLS) at LANL. This research used resources provided by the Los Alamos National Laboratory Institutional Computing Program and Darwin testbed at LANL, which is funded by the Computational Systems and Software Environments subprogram of LANL's Advanced Simulation and Computing program.

Notes and references

1 M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids*, Oxford university press, 2017.
2 M. E. Tuckerman and G. J. Martyna, *J. Phys. Chem. B*, 2000, **104**, 159–178.
3 M. O. Steinhauser and S. Hiermaier, *Int. J. Mol. Sci.*, 2009, **10**, 5135–5216.
4 M. De Vivo, M. Masetti, G. Bottegoni and A. Cavalli, *J. Med. Chem.*, 2016, **59**, 4035–4061.

5 A. P. Thompson, H. M. Aktulga, R. Berger, D. S. Bolintineanu, W. M. Brown, P. S. Crozier, P. J. in't Veld, A. Kohlmeyer, S. G. Moore, T. D. Nguyen, et al., *Comput. Phys. Commun.*, 2022, **271**, 108171.

6 J. Jung, W. Nishima, M. Daniels, G. Bascom, C. Kobayashi, A. Adedoyin, M. Wall, A. Lappala, D. Phillips, W. Fischer, C.-S. Tung, T. Schlick, Y. Sugita and K. Y. Sanbonmatsu, *J. Comput. Chem.*, 2019, **40**, 1919–1930.

7 A. Szabo and N. S. Ostlund, *Modern Quantum Qhemistry: Introduction to Advanced Electronic Structure Theory*, Courier Corporation, 2012.

8 K. Burke, *J. Chem. Phys.*, 2012, **136**, 150901.

9 Y. Zuo, C. Chen, X. Li, Z. Deng, Y. Chen, J. Behler, G. Csányi, A. V. Shapeev, A. P. Thompson, M. A. Wood and S. P. Ong, *J. Phys. Chem.*, 2020, **124**, 731–745.

10 V. L. Deringer, A. P. Bartók, N. Bernstein, D. M. Wilkins, M. Ceriotti and G. Csányi, *Chem. Rev.*, 2021, **121**, 10073–10141.

11 N. Fedik, R. Zubatyuk, M. Kulichenko, N. Lubbers, J. S. Smith, B. Nebgen, R. Messerly, Y. W. Li, A. I. Boldyrev, K. Barros, O. Isayev and S. Tretiak, *Nat. Rev. Chem.*, 2022, **6**, 653–672.

12 M. Kulichenko, J. S. Smith, B. Nebgen, Y. W. Li, N. Fedik, A. I. Boldyrev, N. Lubbers, K. Barros and S. Tretiak, *J. Phys. Chem. Lett.*, 2021, **12**, 6227–6243.

13 O. T. Unke, S. Chmiela, H. E. Sauceda, M. Gastegger, I. Poltavsky, K. T. Schütt, A. Tkatchenko and K.-R. Müller, *Chem. Rev.*, 2021, **121**, 10142–10186.

14 M. Chigaev, J. S. Smith, S. Anaya, B. Nebgen, M. Bettencourt, K. Barros and N. Lubbers, *J. Chem. Phys.*, 2023, **158**, 184108.

15 J. Behler, *J. Chem. Phys.*, 2016, **145**, 170901.

16 H. Ibayashi, T. M. Razakh, L. Yang, T. Linker, M. Olguin, S. Hattori, Y. Luo, R. K. Kalia, A. Nakano, K.-i. Nomura and P. Vashishta, *International Conference on High Performance Computing*, 2023, pp. 223–239.

17 J. Behler and M. Parrinello, *Phys. Rev. Lett.*, 2007, **98**, 146401.

18 A. P. Bartók, M. C. Payne, R. Kondor and G. Csányi, *Phys. Rev. Lett.*, 2010, **104**, 136403.

19 M. Rupp, A. Tkatchenko, K.-R. Müller and O. A. von Lilienfeld, *Phys. Rev. Lett.*, 2012, **108**, 058301.

20 J. S. Smith, O. Isayev and A. E. Roitberg, *Chem. Sci.*, 2017, **8**, 3192–3203.

21 K. T. Schütt, H. E. Sauceda, P.-J. Kindermans, A. Tkatchenko and K.-R. Müller, *J. Chem. Phys.*, 2018, **148**, 241722.

22 N. Lubbers, J. S. Smith and K. Barros, *J. Chem. Phys.*, 2018, **148**, 241715.

23 J. S. Smith, B. Nebgen, N. Mathew, J. Chen, N. Lubbers, L. Burakovsky, S. Tretiak, H. A. Nam, T. Germann, S. Fensin and K. Barros, *Nat. Commun.*, 2021, **12**, 1–13.

24 D. P. Kovács, C. V. D. Oord, J. Kucera, A. E. A. Allen, D. J. Cole, C. Ortner and G. Csányi, *J. Chem. Theory Comput.*, 2021, **17**, 7696–7711.

25 S. Batzner, A. Musaelian, L. Sun, M. Geiger, J. P. Mailoa, M. Kornbluth, N. Molinari, T. E. Smidt and B. Kozinsky, *Nat. Commun.*, 2022, **13**, 1–11.

26 O. T. Unke and M. Meuwly, *J. Chem. Theory Comput.*, 2019, **15**, 3678–3693.

27 K. Yao, J. E. Herr, D. W. Toth, R. Mckintyre and J. Parkhill, *Chem. Sci.*, 2018, **9**, 2261–2269.

28 A. E. Sifain, N. Lubbers, B. T. Nebgen, J. S. Smith, A. Y. Lokhov, O. Isayev, A. E. Roitberg, K. Barros and S. Tretiak, *J. Phys. Chem. Lett.*, 2018, **9**, 4495–4501.

29 T. W. Ko, J. A. Finkler, S. Goedecker and J. Behler, *Nat. Commun.*, 2021, **12**, 398.

30 T. W. Ko, J. A. Finkler, S. Goedecker and J. Behler, *J. Chem. Theory Comput.*, 2023, 3567–3579.

31 M. Eckhoff, K. N. Lausch, P. E. Blöchl and J. Behler, *J. Chem. Phys.*, 2020, **153**, 164107.

32 N. T. P. Tu, N. Rezajooei, E. R. Johnson and C. N. Rowley, *Digit. Discov.*, 2023, **2**, 718–727.

33 J. A. Rackers, L. Tecot, M. Geiger and T. E. Smidt, *Mach. Learn.: Sci. Technol*, 2023, **4**, 015027.

34 E. Shinkle, A. Pachalieva, R. Bahl, S. Matin, B. Gifford, G. T. Craven and N. Lubbers, *J. Chem. Theory Comput.*, 2024, **20**, 10524–10539.

35 S. Magedov, C. Koh, W. Malone, N. Lubbers and B. Nebgen, *J. Appl. Phys.*, 2021, **129**, 064701.

36 I. Batatia, P. Benner, Y. Chiang, A. M. Elena, D. P. Kovács, J. Riebesell, X. R. Advincula, M. Asta, W. J. Baldwin, N. Bernstein, *et al.*, *arXiv*, 2023, preprint, arXiv:2401.00096, DOI: **10.48550/arXiv.2401.00096**.

37 A. E. Allen, N. Lubbers, S. Matin, J. Smith, R. Messerly, S. Tretiak and K. Barros, *npj Comput. Mater.*, 2024, **10**, 154.

38 D. Zhang, H. Bi, F.-Z. Dai, W. Jiang, X. Liu, L. Zhang and H. Wang, *npj Comput. Mater.*, 2024, **10**, 94.

39 A. Jain, S. P. Ong, G. Hautier, W. Chen, W. D. Richards, S. Dacek, S. Cholia, D. Gunter, D. Skinner, G. Ceder and K. A. Persson, *APL Mater.*, 2013, **1**, 011002.

40 R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill and et al., *arXiv*, 2021, preprint, arXiv:2108.07258, DOI: **10.48550/arXiv.2108.07258**.

41 I. Batatia, D. P. Kovacs, G. Simm, C. Ortner and G. Csányi, *Adv. Neural Inf. Process. Syst.*, 2022, **35**, 11423–11436.

42 A. Sriram, A. Das, B. M. Wood, S. Goyal and C. Lawrence Zitnick, *arXiv*, 2022, preprint, arXiv:2203.09697, DOI: **10.48550/arXiv.2203.09697**.

43 F. E. Kelvinius, D. Georgiev, A. P. Toshev and J. Gasteiger, *arXiv*, 2023, preprint, arXiv:2306.14818, DOI: **10.48550/arXiv.2306.14818**.

44 I. Amin, S. Raja and A. Krishnapriyan, *arXiv*, 2025, preprint, arXiv:2501.09009, DOI: **10.48550/arXiv.2501.09009**.

45 S. R. Xie, M. Rupp and R. G. Hennig, *npj Comput. Mater.*, 2023, **9**, 162.

46 G. Hinton, O. Vinyals and J. Dean, *NIPS Deep Learning and Representation Learning Workshop*, 2015.

47 T. Furlanello, Z. Lipton, M. Tschannen, L. Itti and A. Anandkumar, *International Conference on Machine Learning*, 2018, pp. 1607–1616.

48 Y. Yang, J. Qiu, M. Song, D. Tao and X. Wang, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7074–7083.

49 V. Sanh, L. Debut, J. Chaumond and T. Wolf, *arXiv*, 2019, preprint, arXiv:1910.01108, DOI: **10.48550/arXiv.1910.01108**.

50 A. Duval, S. V. Mathis, C. K. Joshi, V. Schmidt, S. Miret, F. D. Malliaros, T. Cohen, P. Liò, Y. Bengio and M. Bronstein, *arXiv*, 2023, preprint, arXiv:2312.07511, DOI: **10.48550/arXiv.2312.07511**.

51 A. Hjorth Larsen, J. Jørgen Mortensen, J. Blomqvist, I. E. Castelli, R. Christensen, M. Dułak, J. Friis, M. N. Groves, B. Hammer, C. Hargus, E. D. Hermes, P. C. Jennings, P. Bjerre Jensen, J. Kermode, J. R. Kitchin, E. Leonhard Kolsbjerg, J. Kubal, K. Kaasbjerg, S. Lysgaard, J. Bergmann Maronsson, T. Maxson, T. Olsen, L. Pastewka, A. Peterson, C. Rostgaard, J. Schiøtz, O. Schütt, M. Strange, K. S. Thygesen, T. Vegge, L. Vilhelmsen, M. Walter, Z. Zeng and K. W. Jacobsen, *J. Phys.:Condens. Matter*, 2017, **29**, 273002.

52 J. Gou, B. Yu, S. Maybank and D. Tao, *arXiv*, 2020, preprint, arXiv:2006.05525, DOI: **10.48550/arXiv.2006.05525**.

53 K. Xu, L. Rui, Y. Li and L. Gu, *European conference on computer vision*, 2020, pp. 664–680.

54 J. Gou, B. Yu, S. J. Maybank and D. Tao, *Int. J. Comput. Vis.*, 2021, **129**, 1789–1819.

55 Y. Chebotar and A. Waters, *Interspeech*, 2016, pp. 3439–3443.

56 S. Gong, Y. Zhang, Z. Mu, Z. Pu, H. Wang, X. Han, Z. Yu, M. Chen, T. Zheng, Z. Wang, et al., *Nat. Mach. Intell.*, 2025, 1–10.

57 S. Matin, E. Shinkle, Y. Pimonova, G. T. Craven, A. Pachalieva, Y. W. Li, K. Barros and N. Lubbers, *Ensemble Knowledge Distillation for Machine Learning Interatomic Potentials*, 2025, **https://arxiv.org/abs/2503.14293**.

58 Q. Xu, Z. Chen, M. Ragab, C. Wang, M. Wu and X. Li, *Neurocomputing*, 2022, **485**, 242–251.

59 L. G. F. dos Santos, B. T. Nebgen, A. E. A. Allen, B. W. Hamilton, S. Matin, J. S. Smith and R. A. Messerly, *J. Chem. Inf. Model.*, 2025, **65**(3), 1198–1210.

60 D. Zhu, Z. Xin, S. Zheng, Y. Wang and X. Yang, *J. Chem. Theory Comput.*, 2024, **20**(13), 5743–5750.

61 J. L. A. Gardner, Z. F. Beaulieu and V. L. Deringer, *Digital Discovery*, 2023, **2**(3), 651–662.

62 G. S. Jung, *J. Chem. Inf. Model.*, 2024, **65**(10), 4797–4807.

63 *ANI-AL data and model GitHub repository*, **https://github.com/atomistic-ml/ani-al**.

64 Y. Liu, X. He and Y. Mo, *arXiv*, 2023, preprint, arXiv:2306.11639, DOI: **10.48550/arXiv.2306.11639**.

65 K. Dang, J. Chen, B. Rodgers and S. Fensin, *Comput. Phys. Commun.*, 2023, **286**, 108667.

66 A. D. Kaplan, R. Liu, J. Qi, T. W. Ko, B. Deng, J. Riebesell, G. Ceder, K. A. Persson and S. P. Ong, *A Foundational Potential Energy Surface Dataset for Materials*, 2025, **https://arxiv.org/abs/2503.04070**.

67 B. Nebgen, N. Lubbers, J. S. Smith, A. E. Sifain, A. Lokhov, O. Isayev, A. E. Roitberg, K. Barros and S. Tretiak, *J. Chem. Theory Comput.*, 2018, **14**, 4687–4698.

68 The hippynn package – a modular library for atomistic machine learning with PyTorch, available online: **https://github.com/lanl/hippynn**.