

# Fine-Tuning Universal Machine-Learned Interatomic Potentials: A Tutorial on Methods and Applications

Xiaoqing Liu

*School of Mathematical Sciences, Shanghai Jiao Tong University, Shanghai 200240, China and  
Shanghai Jiao Tong University-Chongqing Institute of Artificial Intelligence, Chongqing 401329, China*

Kehan Zeng and Zedong Luo

*Shanghai Jiao Tong University-Chongqing Institute of Artificial Intelligence, Chongqing 401329, China*

Yangshuai Wang\*

*Department of Mathematics, National University of Singapore, 10 Lower Kent Ridge Road, Singapore*

Teng Zhao†

*Institute of Natural Sciences, MOE-LSC, and Shanghai National Center for Applied Mathematics,  
Shanghai Jiao Tong University, Shanghai 200240, China and  
Shanghai Jiao Tong University-Chongqing Institute of Artificial Intelligence, Chongqing 401329, China*

Zhenli Xu‡

*School of Mathematical Sciences, MOE-LSC, CMA-Shanghai and Shanghai Center for Applied Mathematics,  
Shanghai Jiao Tong University, Shanghai, 200240, China*

(Dated: August 25, 2025)

Universal machine-learned interatomic potentials (U-MLIPs) have demonstrated broad applicability across diverse atomistic systems but often require fine-tuning to achieve task-specific accuracy. While the number of available U-MLIPs and their fine-tuning applications is rapidly expanding, there remains a lack of systematic guidance on how to effectively fine-tune these models. This tutorial provides a comprehensive, step-by-step guide to fine-tuning U-MLIPs for computational materials modeling. Using the recently released MACE-MP-0 as a representative foundation model, we illustrate the full workflow of dataset preparation, hyperparameter selection, model training, and validation. Beyond methodological guidance, we conduct systematic case studies on solid-state electrolytes, stacking fault defects in metals, semiconductors, solid–liquid interfacial interactions in low-dimensional systems, and more complicated heterointerfaces. These examples demonstrate that fine-tuning substantially improves predictive accuracy while maintaining affordable computational cost, accelerates training convergence, enhances out-of-distribution generalization, and achieves superior data efficiency. Remarkably, fine-tuned foundation models can even capture aspects of long-range physics without explicit corrections. Together, these results highlight that fine-tuning not only provides a practical recipe for applying U-MLIPs, but also offers new insights into their physical fidelity and potential for advancing large-scale atomistic simulations. To support practical applications, we include code examples that enable researchers, particularly those new to the field, to efficiently incorporate fine-tuned U-MLIPs into their workflows.

## I. INTRODUCTION

Machine-learned interatomic potentials (MLIPs) have become an essential tool in materials modeling, offering near-density functional theory (DFT) accuracy at significantly reduced computational cost [1–14]. A broad range of MLIP frameworks have been developed in recent years, including neural network potentials [1, 4], kernel-based approaches [2, 5], and equivariant graph neural networks [6, 7], each demonstrating excellent performance across diverse materials systems. These models enable accurate and large-scale atomistic simulations, and their

integration with simulation platforms such as ASE [15] and LAMMPS [16] has facilitated their widespread adoption in the field. For comprehensive reviews on MLIPs, the reader is referred to Refs. [17–21].

Traditionally, most MLIPs are trained on narrowly defined chemical or structural domains tailored to specific applications. While such models can achieve high accuracy within their training scope, they often lack transferability to systems beyond that domain. In response, there has been growing interest in a new class of models, often referred to as general-purpose, foundation, or universal MLIPs (U-MLIPs) [22–27]. Inspired by foundation models in natural language processing and computer vision [28], these models are pre-trained on large and diverse materials datasets [29–31], aiming to provide broad coverage across the periodic table and structural configuration space. They offer strong baseline performance and could be readily applied to new systems with min-

---

\* [yswang@nus.edu.sg](mailto:yswang@nus.edu.sg)

† [zhaoteng\\_sjtu@sjtu.edu.cn](mailto:zhaoteng_sjtu@sjtu.edu.cn)

‡ [xuzl@sjtu.edu.cn](mailto:xuzl@sjtu.edu.cn)

imal additional training (i.e., fine-tuning), substantially lowering the barrier to entry for MLIP applications in materials science. In Table I, we summarize representative U-MLIPs, along with their publication year, underlying architectures, and training data characteristics.

Among existing U-MLIPs, MACE-MP-0 stands out as a near state-of-the-art model, demonstrating good accuracy and robust stability in molecular dynamics simulations across a wide range of applications [22]. It serves as the central focus of this work. MACE-MP-0 is built upon the MACE architecture [6], which integrates the atomic cluster expansion (ACE) formalism [5, 37] with efficient tensor decomposition techniques [38] and higher-order equivariant message passing. Leveraging this design, MACE-MP-0 is pre-trained on a large dataset from the Materials Project, and achieves strong performance across diverse materials benchmarks. Several subsequent variants of the model have been introduced recently to address specific issues such as repulsion, phonons, and high-pressure stability, further broadening its applicability.

While MACE-MP-0 and its variants have demonstrated robust performance in molecular dynamics simulations [22], its predictive accuracy—like that of other U-MLIPs—remains limited for certain tasks, particularly in modeling mechanical properties such as elastic constants and stacking fault energies in elemental alloys [39]. This underscores the need for fine-tuning, which adapts pre-trained models to specific systems by refining their parameters using high-fidelity computational or experimental data. Although the fine-tuning of atomistic foundation models has garnered growing attention, comprehensive and systematic studies remain limited, especially practical tutorials that guide beginners through the fine-tuning process. Recent works have started addressing this gap [40–48], and continued efforts are expected to clarify best practices across various application scenarios. While the number of available U-MLIPs and their fine-tuning applications is rapidly expanding, there remains a lack of systematic guidance on how to effectively fine-tune these models, which serves as the primary motivation of this tutorial.

Beyond the pursuit of higher accuracy, a central motivation for fine-tuning in the context of MLIPs lies in a fundamental question: given a fixed dataset, should one fine-tune an existing foundation model or train a new model from scratch? In practice, fine-tuning is often more efficient in terms of training time and data usage, and in many cases yields comparable or even superior accuracy. While foundation models, pre-trained on large and diverse datasets, effectively capture general chemical and structural patterns, they often lack the precision required for specific systems. Fine-tuning addresses this limitation by incorporating system-specific data, thereby enhancing accuracy with significantly lower data and computational costs. Moreover, fine-tuned models typically converge faster due to the advantage of informed initialization. Therefore, when a suitable foundation model

is available, fine-tuning presents a compelling alternative to training-from-scratch.

This tutorial provides a practical guide to fine-tuning U-MLIPs, aiming to support researchers—especially those new to the field—by filling the current gap in systematic guidance for fine-tuning foundation models. The goal is to provide step-by-step instructions for preparing data, training models, selecting hyperparameters, and applying the resulting potentials in various atomistic simulations. To keep the presentation accessible, theoretical details are kept to a minimum, with references provided for readers seeking a deeper mathematical and physical understanding. The methods introduced here are demonstrated using widely used tools such as ASE, LAMMPS, and the Random Batch Molecular Dynamics (RBMD) interface [49]. MACE-MP-0 is selected as the main example due to its broad applicability, high accuracy across benchmark tasks, and demonstrated stability in molecular dynamics simulations [22].

Beyond methodological guidance, we conduct extensive numerical experiments that yield several key insights. Fine-tuning consistently improves predictive accuracy and accelerates convergence while maintaining affordable computational cost. It enhances out-of-distribution generalization and achieves higher data efficiency compared to training from scratch, even when using reduced datasets. Interestingly, fine-tuned foundation models sometimes show improved accuracy even for cases involving long-range effects, suggesting that certain aspects of such interactions may already be partially captured during pre-training. Taken together, these findings show that fine-tuning not only offers a practical recipe for applying U-MLIPs, but also provides deeper insights into their physical fidelity and their potential to advance large-scale atomistic simulations.

It is worth noting that the fine-tuning of MACE-MP-0 can be efficiently carried out using the RBMD package, which enables large-scale particle simulations at the nano- and microscale. In contrast to conventional molecular dynamics frameworks, RBMD leverages random batch algorithms [50–53] to efficiently handle non-bonded interactions, supporting simulations of up to 10 million particles on a single GPU with a CPU core [49]. Moving forward, we aim to integrate the RBMD platform with MACE-based MLIPs for more practical and large-scale simulations. To support this integration, an open-access platform is available at <https://www.randbatch.com/rbmd>, where further examples and implementation scripts for realistic simulation scenarios will be released in the future.

The remainder of this tutorial is organized as follows. In Section II, we introduce the overall fine-tuning workflow for U-MLIPs, including a brief overview of the MACE-MP-0 framework, data preparation, potential training, and model validation. Section III presents six illustrative applications of fine-tuning U-MLIPs: (i) force accuracy evaluation in the solid-state electrolyte  $\text{Li}_{10}\text{GeP}_2\text{S}_{12}$  (LGPS), (ii) stacking fault energetics in

TABLE I. List of the representative U-MLIPs.

Model Name	Year	Architecture	Data Description
M3GNET [27]	2022	SchNet [32]	Materials Project (MP), 89 elements, 62783 compounds
CHGNet [23]	2023	GNN + Charge	MP + Trajectory, 89 elements, 146000 compounds
ALIGNN-FF [26]	2023	Line GNN	JARVIS-DFT, 72708 compounds
PFP (Matlantis) [33]	2023	Tensorial GNN	Custom, $\approx$ 10 million configurations
GNoME [24]	2023	NequIP [8]	MP + Custom, $\approx$ 89 million configurations
DPA-1 [25], DPA-2 [25]	2023, 2024	DeepMD [4]	Open-access datasets including Alloy, OC2M, and others.
<b>MACE-MP-0</b> [22]	2024	MACE [6]	Same data as used to build CHGNet
SevenNet-0 [34]	2024	NequIP [8]	Same data as used to build CHGNet
MatterSim [35]	2024	Graph transformer	MP+Alexandria+Custom, $\approx$ 17 million configurations
EquiformerV2-OMAT24 [31]	2024	EquiformerV2 [36]	MP+Alexandria+Custom, $\approx$ 118 million configurations

body-centered cubic molybdenum (Mo), (iii) generalization tests on out-of-distribution in semiconductor silicon, (iv) surface interactions at the graphene–water interface, (v) molecule–oxide interactions, and (vi) solid–solid interfaces, the latter two designed to assess the ability of fine-tuned models to capture long-range effects. Finally, Section IV provides a summary and outlook on the development and fine-tuning of U-MLIP models.

## II. METHODS

In this section, we introduce the overall procedure for fine-tuning U-MLIPs, using MACE-MP-0 as a representative model. An overview of the fine-tuning workflow is shown in Figure 1. The section is organized as follows: Section II A briefly describes the MACE-MP-0 model, common fine-tuning practices, and recent implementation updates. Dataset construction, a critical component of MLIP development, is discussed in Section II B. Section II C outlines the fine-tuning process, including hyperparameter choices. Model validation is addressed in Section II D, which is a prerequisite for reliable molecular dynamics simulations.

The fine-tuning procedure is supported by the official MACE repository: <https://github.com/ACEsuit/mace>. Installation instructions are also available in the tutorial repository: <https://github.com/John2021-hub/mace-ft-tutorial.git>. The RBMD interface [49] supports both fine-tuning and molecular dynamics simulations with MACE-based MLIPs.

### A. The MACE-MP-0 Model: Architecture, Fine-Tuning, and Recent Developments

We begin by reviewing the MACE-MP-0 model in Section II A 1. The fine-tuning procedures introduced here are broadly applicable to other U-MLIPs with only minor adjustments.

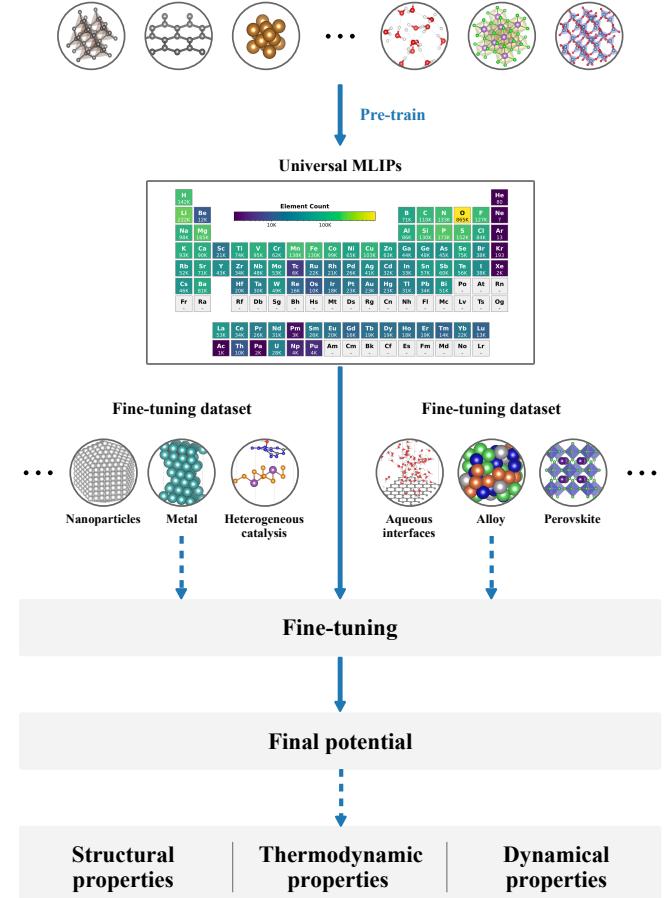


FIG. 1. Overview of the fine-tuning workflow

#### 1. Overview of the MACE-MP-0 Model

Before introducing U-MLIPs, we briefly recall the concept of MLIPs. These models map atomic positions and chemical elements to the potential energy of a given atomic system. The MACE architecture [6] is a representative example, extending the Atomic Cluster Ex-

pansion (ACE) framework [5, 37, 54], which was originally developed for MLIPs and has since found applications beyond atomistic modeling [55–58]. MACE is an equivariant message-passing graph tensor network [59, 60], where many-body atomic geometry is encoded layer by layer. Messages are built from linear combinations of tensor product bases derived from two-body permutation-invariant polynomials in a spherical basis [5, 37]. Equivariance is maintained through tensor contractions with irreducible representations and generalized Clebsch–Gordan coefficients [38, 61]. The final output represents the per-atom contribution to the total potential energy. Further architectural details are provided in Ref. [6].

The foundation model MACE-MP-0, trained on the Materials Project dataset [62], has demonstrated strong performance in a range of applications [22], particularly in enabling stable molecular dynamics simulations across diverse chemical environments. It is released in three variants, differentiated by increasing levels of message-passing equivariance ( $L = 0, 1, 2$ ). A subsequent revision, MACE-MP-0b, introduced minor improvements such as refined pairwise repulsion and improved treatment of isolated atoms. Further variants have been developed to address specific limitations: MACE-MP-0b2 enhances stability under high pressure, MACE-MP-0b3 corrects phonon behavior, and MACE-MPA-0 and MACE-OMAT-0 extend training coverage. All models are publicly available at <https://github.com/ACEsuit/mace-mp/>. In this work, we adopt MACE-MP-0b3 as the base model for fine-tuning. For simplicity, we continue to refer to it as MACE-MP-0 unless otherwise stated.

Despite the general robustness of these models, stable MD simulations do not guarantee accurate predictions of physical properties. Fine-tuning remains essential for adapting U-MLIPs to specific systems. However, to the best of our knowledge, there is currently no systematic tutorial guiding users through the fine-tuning of U-MLIPs for practical applications. Addressing this gap serves as the central motivation for the present work.

## 2. Fine-Tuning Strategies

Fine-tuning pre-trained models is widely adopted to improve accuracy and transferability in various domains, including image analysis [63, 64] and natural language processing [65–67]. In molecular modeling, fine-tuning typically follows a predictor–corrector paradigm: a U-MLIP serves as an initial predictor that provides a robust baseline for atomistic simulations. The model is then fine-tuned on a system-specific dataset, allowing it to correct and adapt to the target system with improved accuracy.

Multi-head replay fine-tuning extends this idea by attaching multiple task-specific output heads to a shared pre-trained backbone [68]. Each head is tailored to a different target system or property, while the backbone cap-

tures general representations. By partially freezing the backbone, this approach enables efficient training across multiple tasks and reduces redundancy. This design also mitigates catastrophic forgetting by preserving the performance of previously trained heads while adapting to new tasks. An implementation of this method is available at <https://github.com/ACEsuit/mace/tree/main>.

From an optimization perspective, existing fine-tuning strategies for U-MLIPs can be broadly categorized into: (i) Full-parameter fine-tuning, where all backbone and head parameters are updated [69]. This maximizes task-specific adaptability but incurs higher computational cost and greater risk of overfitting, especially when the target dataset is small. (ii) Partial freezing, where early layers or invariant feature extractors are kept fixed while only later layers or heads are updated. This reduces the number of trainable parameters, leading to faster convergence and better knowledge retention from the pre-trained model, but may limit adaptation to systems with strong distribution shifts [70]. (iii) Lightweight adaptation layers (e.g., low-rank adapters or bias tuning), which introduce a small set of trainable parameters on top of the frozen backbone [71]. This approach offers a favorable trade-off between efficiency and accuracy, and has shown promise in retaining generalization while improving data efficiency.

In molecular modeling, the choice of strategy should be guided by the similarity between the pre-training and target domains, the size and diversity of the fine-tuning dataset, and the computational budget. A comprehensive benchmark to determine the optimal strategy under different scenarios is still lacking, but future work will aim to provide such guidance.

## 3. Recent Updates

This section briefly introduces two recent updates relevant to U-MLIP deployment: the fast equivariant kernel `cuEquivariance`, and an implementation of long-range dispersion corrections with efficient neighbor list handling. These tools are integrated into our examples (cf. Section III) to demonstrate the fine-tuning and application of the MACE-MP-0 model in practical simulations.

*a. The fast equivariant kernel `cuEquivariance`.* Equivariance, referring to the preservation of pre-defined physical symmetry under group operations such as rotations and translations, is a key property of physically meaningful interatomic potentials. Models that are naturally equivariant are typically more data-efficient, possess better physicality and exhibit improved generalization.

`cuEquivariance` is an NVIDIA-developed library for building high-performance equivariant neural networks based on segmented tensor products within easily accessible Python interface. It provides a flexible API and optimized CUDA kernels, with bindings for both PyTorch and JAX. The code is available at <https://github.com/>

NVIDIA/cuEquivariance.

Fine-tuned MACE-MP-0 models (cf. Section II C) can be integrated with cuEquivariance in Python-based simulation environments such as ASE [15], enabling significantly faster inference—typically by a factor of 3 to 10. Example usage and performance benchmarks are provided in Section III A. The following code snippet illustrates how to activate the fast kernel via a simple keyword:

```
1 from mace.calculators import MACECalculator
2 mace_calc =
    MACECalculator(model_paths="MACE.model",
    enable_cueq=True, device="cuda")
```

Listing 1. Enable cuEquivariance kernel

As reported by developers, the jax implementation is expected to have at least extra few factors of speed up further.

*b. Long-range dispersion corrections with efficient neighbor list handling.* Frequent neighbor list updates can introduce significant computational overhead, particularly when large cutoffs are used for dispersion corrections such as DFT-D3 [72]. To mitigate this, a simple yet effective neighbor list implementation with a skin layer is provided in ASE. This approach reduces the frequency of neighbor list reconstruction and improves overall computational efficiency.

The implementation allow similar update strategy in LAMMPS and enables the reuse of neighbor lists across multiple simulation steps. An example configuration is shown below:

```
1 calc = TorchDFTD3Calculator(atoms=atoms,
    device="cuda", damping="bj",
    every=2, delay=10, check=True, skin=2.0)
```

Listing 2. Enable neighbor list reuse

In this example, the parameters `every`, `delay`, `check`, and `skin` control the update frequency and reuse behavior of the neighbor list:

- `every`: interval between neighbor list updates.
- `delay`: buffer before enforcing the next update.
- `check`: whether to validate the reuse condition.
- `skin`: width of the buffer zone (in Å), allowing small displacements without triggering a rebuild.

The implementation is available at: <https://github.com/CheukHinHoJerry/torch-dftd>.

## B. Data Preparation

The quality of a MLIP is closely tied to the quality of the reference dataset. MLIPs are typically trained via supervised regression on a large set of atomic configurations, each labeled with total energies, atomic forces, and often stress tensors, usually computed from DFT. The

same principles apply to fine-tuning U-MLIPs, where a well-designed dataset is crucial for achieving improved accuracy and transferability [40, 42, 43, 73, 74].

### 1. Empirical Dataset Construction

Empirical dataset construction relies on physically motivated or experience-driven methods. These include using existing public datasets, generating configurations via random perturbations, or sampling configurations from molecular dynamics (MD) or Monte Carlo (MC) simulations. Each method offers distinct advantages and is best suited for particular types of systems or user expertise. In the following, we briefly describe each approach and offer practical recommendations based on our experience.

*a. Using publicly available datasets.* Depending on observables or quantities of target system, open-access datasets that have been developed for machine-learned interatomic potentials can be readily used for fine-tuning U-MLIPs [29–31, 75–77]. A key consideration is format compatibility. For example, the MACE-MP-0 model expects data in the .xyz format, whereas datasets for Deep Potential (DP) models [4] are typically provided in .npy format. To address this, we provide a convenient script that converts .npy files into .xyz files:

```
1 python transfer_npy_xyz.py train.npy train.xyz
```

Listing 3. Convert .npy data to .xyz

Another consideration is dataset size. Public datasets designed for training-from-scratch are often much larger than what is needed for fine-tuning. Since U-MLIPs already encode broad chemical and structural trends, effective fine-tuning can usually be achieved with significantly fewer configurations. This observation will be demonstrated in a case study in Section III A. We note that using publicly available datasets is especially recommended for beginners, as it requires minimal setup and provides immediate access to high-quality reference data.

*b. Random perturbation of structures.* Another common approach is to apply random displacements to atoms in relaxed structures to sample configurations near local energy minima (local equilibrium). This method is simple and widely used, especially for systems with crystalline structures. If stress data are required, small random deformations of the simulation cell (i.e., strain perturbations) should also be applied. Random perturbations can be generated using ASE’s `rattle` function, typically with Gaussian-distributed displacements of 0.1–1.0 Å. Care must be taken to avoid unphysical structures with atoms too close together. This issue can be mitigated by applying a Monte Carlo filtering procedure that enforces a minimum interatomic distance, where configurations with excessively short atomic separations are accepted with low probability [78]. This method is recommended for users who are familiar with atomistic simulation workflows and have sufficient computational resources. It is particularly well suited for

advanced users with prior knowledge of the target system. Notably, when combined with the filtering techniques discussed later, well-designed perturbation strategies can produce compact and physically meaningful datasets that are particularly effective for fine-tuning U-MLIPs [79].

*c. Sampling from MD or MC simulations.* When no well-defined structural prototype exists, as in amorphous phases or liquids, perturbation methods are not suitable. Instead, ab initio molecular dynamics [80, 81] or Monte Carlo simulations provide a more general approach for generating configurations. Snapshots are extracted at regular intervals along the trajectory to build the dataset. Although computationally more demanding, this approach is essential when structural templates are unavailable. To further improve dataset diversity and relevance for fine-tuning, perturbations are often applied to MD-generated structures, and in some cases, MD simulations with enhanced sampling are employed [82, 83].

## 2. Uncertainty-Based Dataset Construction

Manual dataset design often depends on expert intuition, which may introduce inductive biases, redundant structures, or sampling imbalances. Such issues can undermine data efficiency and hinder the generalizability of trained MLIPs. As a remedy, uncertainty-guided data selection has emerged as a systematic and efficient alternative for both training MLIPs and fine-tuning U-MLIPs [84–89].

Uncertainty quantification (UQ) estimates the confidence of a model’s predictions and can be used to identify regions in configuration space where the model is unreliable. Incorporating such estimators enables active learning strategies that prioritize configurations with high predictive uncertainty, thereby improving accuracy while minimizing data redundancy.

Several definitions and methodologies for uncertainty estimation are commonly employed in the context of MLIPs. These approaches can be broadly categorized into frequentist and Bayesian frameworks, as summarized in Table II. Frequentist techniques include model ensembles [90] and feature-space distance metrics [91], while Bayesian methods rely on posterior inference [87], dropout-based approximations [88], or analytical forms available in tractable models such as Gaussian processes and linear regressions [79]. In addition, hybrid approaches have been developed to separately quantify aleatoric (data-inherent) and epistemic (model-related) uncertainties [92]. A comprehensive discussion of UQ in MLIPs is beyond the scope of this tutorial. However, we emphasize that this remains an active area of research, and we refer interested readers to recent reviews for further insight [93–95].

*a. Filtering a candidate set.* In cases where a large pool of configurations is available (e.g., from long MD simulations or existing datasets), uncertainty-based fil-

tering can be applied to construct an efficient fine-tuning set. Here, an uncertainty estimator (such as an ensemble or a surrogate model with Bayesian regression) is used to assign a confidence value to each configuration. Configurations with higher estimated uncertainty are prioritized, as they are more likely to contribute new information to the model. This strategy has been used successfully in recent works, such as Ref. [79], where a linear ACE model is trained on top of an existing representation to guide uncertainty filtering. We evaluate the data efficiency of fine-tuning by comparing it against uncertainty-based filtering in Section III B.

*b. Active learning strategy.* Active learning provides a fully iterative framework for dataset construction. Starting from a small initial set of labeled configurations, the MLIP is trained and then used to run MD simulations. Configurations from these simulations are evaluated by a UQ metric, and the most uncertain ones are selected for DFT labeling. The newly labeled data are added to the training set, and the process is repeated until convergence or a stopping criterion is met (e.g., error saturation or maximum data budget). This strategy is particularly effective for exploring diverse chemical spaces or unknown physical regimes.

While uncertainty-driven methods offer significant advantages, they are not universally applicable. Analytical UQ is often limited to linear models or kernel-based methods such as Gaussian process regression. For large, non-linear models like deep neural networks, approximate UQ methods (e.g., ensembles, dropout) may require careful calibration [105, 106]. Additionally, active learning workflows introduce algorithmic and software complexity, which may be a barrier for new users.

For novice users, we recommend the following practical guidelines: (i) Small, chemically narrow systems: Start with a simple ensemble-based UQ (e.g., multiple MACE models trained from different random seeds) and select new configurations with the highest predicted force uncertainty. (ii) Chemically diverse or poorly explored systems: Consider dropout-based UQ during inference to reduce training cost while still capturing model variance, but apply a post-hoc calibration step such as isotonic regression or temperature scaling. (iii) When DFT labeling is expensive: Limit the number of configurations per iteration by combining UQ-driven selection with structural diversity filters (e.g., farthest-point sampling in descriptor space [107]). These approaches balance exploration efficiency, calibration accuracy, and computational feasibility, enabling active learning workflows that are both effective and accessible to new users.

## C. Fine-Tuning

This section outlines the main steps and considerations for fine-tuning the MACE-MP-0 model. Fine-tuning refers to updating the parameters of a pre-trained U-MLIP on a task-specific dataset, while keeping the

TABLE II. Representative uncertainty quantification (UQ) methods used in MLIPs.

Method	Key Idea	Category
Ensemble [96]	Use variance across multiple independently trained models	Frequentist
Dropout UQ [97]	Dropout at inference to sample pseudo-ensembles	Bayesian
Bayesian NN [98]	Infer posterior over weights via VI [99] or MCMC [100]	Bayesian
Evidential UQ [101]	Predict mean and variance directly as outputs	Evidential
GPR variance [102]	Analytical uncertainty from Gaussian processes	Bayesian
Feature distance [103]	Distance to nearest training point in feature space	Geometric
Aleatoric + Epistemic [104]	Explicitly model data and model uncertainty	Combined

overall model architecture fixed. This strategy leverages the broad physical knowledge encoded in the foundation model and enables efficient adaptation to new systems with relatively limited data.

A typical fine-tuning workflow involves constructing the training dataset, (and usually a validation dataset) set with the above mentioned dataset construction methods. Table III summarizes representative hyperparameters and their recommended ranges, consistent with the example script provided in Listing 4.

Listing 4 presents a fine-tuning script using the `run_train.py` utility provided by the MACE package:

```

1 python3 $MACE_DIR/mace/cli/run_train.py \
2   --name=$RES_DIR \
3   --foundation_model="XXX.model" \
4   --model_dir=results/$RES_DIR \
5   --log_dir=results/$RES_DIR \
6   --checkpoints_dir=results/$RES_DIR \
7   --results_dir=results/$RES_DIR \
8   --train_file="train.xyz" \
9   --valid_file="valid.xyz" \
10  --energy_weight=1.0 \
11  --forces_weight=10.0 \
12  --stress_weight=1.0 \
13  --loss="universal" \
14  --forces_key=dft_force \
15  --energy_key=dft_energy \
16  --stress_key=dft_stress \
17  --lr=0.0005 \
18  --scaling="rms_forces_scaling" \
19  --batch_size=4 \
20  --max_num_epochs=150 \
21  --ema \
22  --ema_decay=0.99 \
23  --weight_decay=1e-6 \
24  --amsgrad \
25  --default_dtype="float64" \
26  --clip_grad=10 \
27  --device=cuda \
28  --seed=$SEED \
29  --num_samples_pt=500 \
30  --swa_energy_weight=100.0 \
31  --swa_forces_weight=10.0 \
32  --swa_stress_weight=1.0 \
33  --swa \
34  --swa_lr=5e-4 \
35  --E0s="{3:-0.29745415, 15:-1.88719893,
16:-1.08090657, 32:-0.77926738}" \

```

Listing 4. Fine-tuning template for the MACE-MP-0 model.

From our experience, a medium-sized model ( $L = 1$ ) provides the most balanced trade-off between accuracy and computational efficiency. While the small model ( $L = 0$ ) is computationally inexpensive but lacks expressivity, the large model ( $L = 2$ ) increases accuracy

at the cost of significantly reduced efficiency. The following recommendations are in general helpful on stability and accuracy of the resulting model from fine-tuning:

- Isolated atom energies (E0s): It is very important to compute your own E0s, which specify the isolated atom energy as a dictionary in your DFT calculation rather by estimating it from least square regression with `E0s="average"`. When computing your E0s, use spin polarized calculations. You may also use `E0s="foundation"` to use the same E0s when using exactly the same DFT settings and software as Materials Project.
- Learning rate (lr): If the training or validation loss stagnates or diverges, consider modifying the learning rate from recommended range for more stable convergence.
- Loss weights: Depending on the target properties, modify the ratio between energy, forces and stress weight in fine-tuning within the recommended range.
- Cutoff radius (r\_max): Do not modify the pre-defined cutoff radius that the foundation model used.
- Batch size: A larger batch size generally improves training stability but requires more GPU memory. Batch size larger than 20 is not recommended since a larger batch size, in general, tend to deteriorate the generalizability of machine learning models.
- EMA and SWA: Enable exponential moving average (`ema`) and stochastic weight averaging (`swa`) for smoother convergence and better generalization. If training dynamics appear unstable, set `swa` to 0.99999. The model with best validation error from either `pre-swa` or `post-swa` based (i.e. `stageone` and `stagetwo` in Section II D) on validation loss are outputted to be further verified on other properties.
- Number of samples in the pretrained head (`num_samples_pt`): This parameter controls how many samples from the pretrained model's output

TABLE III. Representative hyperparameters for fine-tuning MACE-MP-0. The recommended values are representative for the cases considered in this work, and may require adjustment for different systems.

Parameter	Description	Recommended Range
<code>energy_weight</code>	Energy loss weight	1.0 – 20.0
<code>forces_weight</code>	Force loss weight	1.0 – 20.0
<code>stress_weight</code>	Stress loss weight	0.0 – 20.0
<code>lr</code>	Learning rate	$10^{-4} – 10^{-3}$
<code>batch_size</code>	Batch size	2 – 20
<code>ema_decay</code>	EMA decay rate	0.98 – 0.99999
<code>swa</code>	Enable stochastic weight averaging	True / False
<code>swa_lr</code>	Learning rate for SWA phase	$10^{-5} – 10^{-4}$

distribution are retained for the fine-tuning stage. A larger value preserves a richer set of features and variance from the pretrained head, which can improve MD stability. In other words, it acts as a form of “knowledge retention” that mitigates catastrophic forgetting of the pretrained model. However, there is a trade-off: if it is too large, the finetuned model remains overly biased towards the pretrained distribution, which may limit its ability to fit task-specific data and thus cap the achievable accuracy.

We recommend monitoring convergence behavior (cf. Fig. 2) and validation accuracy throughout the fine-tuning process. Evaluating model performance on unseen configurations is crucial and will be discussed in the next subsection V

#### D. Model Validation and Evaluation

After completing the fine-tuning process, several output files are generated by the training script. Among them, the most critical is the log file, which contains all essential information needed to reproduce the results. This includes the selected hyperparameters, loss trajectories over all training epochs, and final test errors (typically reported in terms of RMSE). Toward the end of the log, a summary table presents test errors for energies, forces, and—if available—stresses.

Additionally, the script automatically generates diagnostic plots that visualize the training and validation loss curves, as well as scatter plots comparing predicted and reference energies and forces. These visualizations provide a qualitative assessment of model performance. In particular, smooth loss convergence and tightly clustered scatter plots are indicative of successful fine-tuning.

As an illustration, we refer to the example in Section III A, where such diagnostic plots are used to evaluate the performance of a fine-tuned model. Figure 2 displays the evolution of training and validation losses, along with the final energy and force scatter plots.

The output directory also contains several key model files:

- `MACE-FT.model`: the complete fine-tuned model used for inference;
- `MACE-FT.compiled.model`: a compiled version optimized for deployment;
- `MACE-FT.stageone.model` and `MACE-FT.stagetwo.model`: intermediate models from different training stages (e.g., before and after stochastic weight averaging).

These models can be directly used in downstream atomistic simulations and analysis workflows to evaluate a variety of important material properties. These include structural and mechanical properties such as bulk modulus and defect formation and migration energies, thermodynamic properties such as density, pressure, and phonon spectra, as well as dynamical properties such as radial distribution functions and diffusion coefficients obtained from molecular dynamics simulations.

a. *Example: Structural Relaxation with ASE.* Once fine-tuned, the model can be seamlessly integrated into the Atomic Simulation Environment (ASE) for structural optimization and property evaluation. The example below demonstrates how to perform structural relaxation on a given atomic configuration using the `MACECalculator` within ASE:

```

1  from mace.calculators import MACECalculator
2  ase_atoms.calc = MACECalculator("MACE-FT.model")
3
4  from ase.optimize.precon import PreconLBFGS
5  optimizer = PreconLBFGS(ase_atoms,
6    variable_cell=True)
7  optimizer.run(fmax=1e-5)

```

Listing 5. Structural relaxation using a fine-tuned MACE model

This procedure minimizes both atomic positions and cell parameters using the preconditioned L-BFGS algorithm. The same workflow can be extended to defect formation energy calculations, nudged elastic band (NEB) transition state searches, or molecular dynamics (MD) simulations.

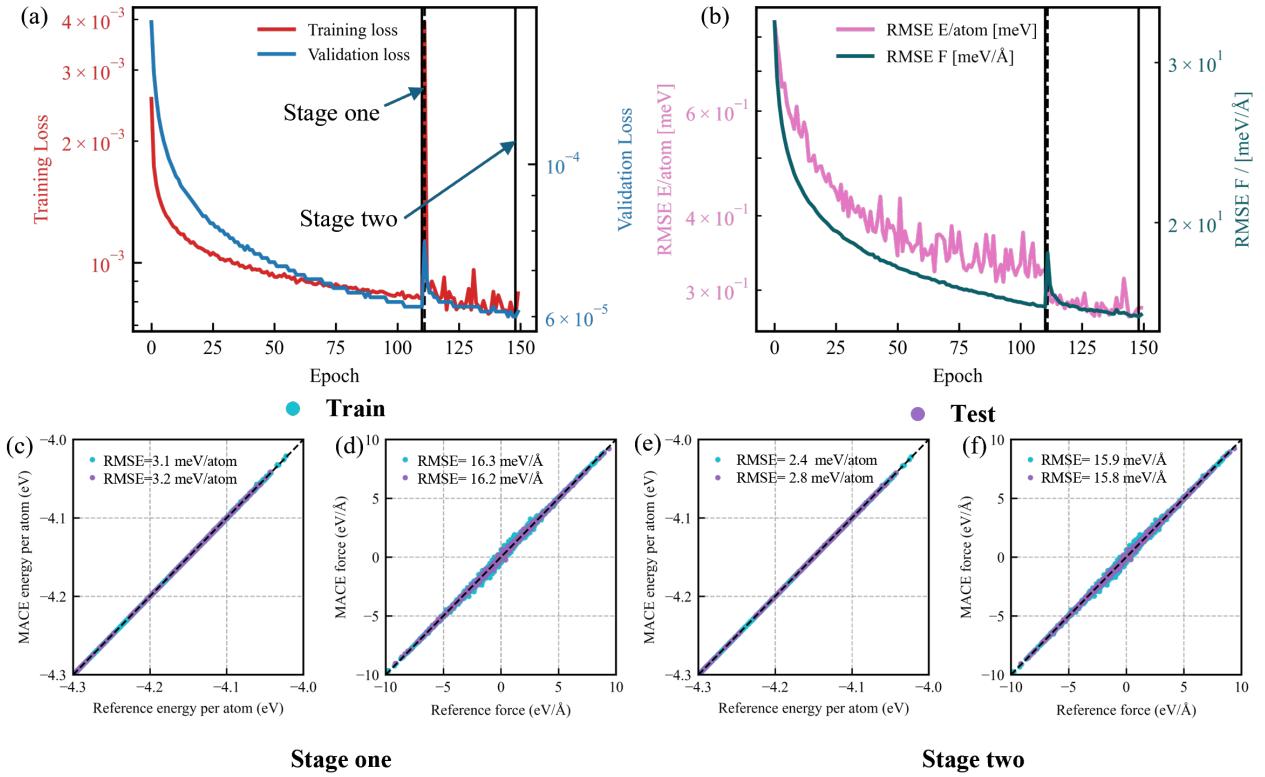


FIG. 2. Training and validation loss curves, and fitting accuracy for energy and force predictions. The left panel corresponds to the initial training stage (before stochastic weight averaging), while the right panel shows results after SWA.

b. *Example: Using the Model in LAMMPS.* To use the fine-tuned model in LAMMPS, it must first be converted to a LAMMPS-compatible format using the provided utility script:

```
1 python
<mace_repo_dir>/mace/cli/create_lammps_model.py
MACE-FT.model
```

Listing 6. Exporting the model for LAMMPS

Executing the above command generates a LAMMPS-compatible model file named `MACE-FT.model-lammps.pt`, which can be directly used in LAMMPS simulations via the following commands:

```
1 pair_style mace no_domain_decomposition
2 pair_coeff * * MACE-FT.model-lammps.pt element1
element2 element3
```

Listing 7. Using the fine-tuned MACE potential in LAMMPS

Here, `pair_style` specifies the MACE interaction style without domain decomposition, and `pair_coeff` is used to load the compiled model file along with the corresponding atomic species. In our molecular dynamics simulations, we adopted the `metal` unit style and applied a canonical (NVT) ensemble to maintain the temperature at 298 K.

We refer readers to our third case study (Section III D) for a complete example of running MD sim-

ulations in LAMMPS using a fine-tuned model. All input files and scripts are available in our public tutorial repository: <https://github.com/John2021-hub/mace-ft-tutorial.git>.

### III. EXAMPLES AND APPLICATIONS

The fine-tuned models integrate seamlessly with widely used atomistic simulation platforms such as ASE, LAMMPS and RBMD (see Appendix VI), enabling efficient molecular dynamics and other atomistic simulations. In this section, we illustrate the effectiveness of fine-tuning using representative case studies on solid-state electrolytes, metallic stacking faults, and interfacial phenomena in low-dimensional systems. The pretrained foundation models are publicly available at <https://github.com/ACESuit/mace-mp/>. In addition, we provide fine-tuning scripts, example outputs, and best practices in our tutorial repository: <https://github.com/John2021-hub/mace-ft-tutorial.git>.

We performed fine-tuning and molecular dynamics simulations on a single NVIDIA A800 GPU with 80 GB of HBM memory. The software environment was configured with CUDA 11.7 and the NVIDIA HPC SDK (nvhpc) 20.9. Unless otherwise specified, all training runs were carried out under this single-GPU setup.

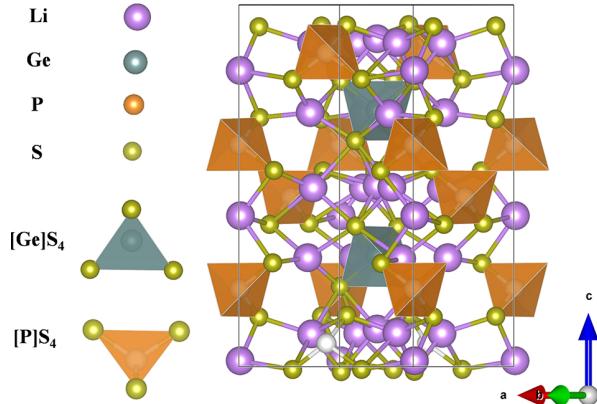


FIG. 3. Atomic structure of  $\text{Li}_{10}\text{GeP}_2\text{S}_{12}$ , a solid-state electrolyte with high lithium-ion conductivity.

We present four representative examples to illustrate the fine-tuning of U-MLIPs across different materials contexts. Solid-state electrolytes are chosen to assess improvements in force prediction essential for accurate ion-transport simulations. Stacking fault defects in metals are included to evaluate generalization to defect configurations and the accurate prediction of stacking fault energies. Silicon with a diverse and complex dataset is used to evaluate out-of-distribution generalization. Solid–liquid interfacial systems are examined to test the model’s ability to capture interfacial energetics critical to phase transitions and growth processes.

#### A. $\text{Li}_{10}\text{GeP}_2\text{S}_{12}$

In this example, we demonstrate the fine-tuning procedure on a solid-state electrolyte system, lithium germanium phosphorous sulfide ( $\text{Li}_{10}\text{GeP}_2\text{S}_{12}$ , abbreviated as LGPS), using a dataset published on DPAISquare [75]. LGPS is a prototypical member of the LGPS-type thio-phosphate family, and has attracted significant attention due to its high lithium-ion conductivity, making it a promising candidate for next-generation all-solid-state batteries. Modeling such materials requires accurate interatomic potentials capable of describing both local structural features and ionic transport mechanisms, making LGPS an ideal testbed for evaluating the transferability and fine-tuning of U-MLIPs.

The LGPS dataset published on DPAISquare [75] can be converted into the .xyz format supported by MACE, making it directly usable in the fine-tuning workflow (cf. Listing 3). Figure 3 shows the atomic structure of a typical configuration from this dataset.

Table IV summarizes the test accuracy of multiple MACE foundation models after fine-tuning. All models achieve comparable accuracy in terms of root-mean-square error (RMSE) with respect to DFT reference values. The two rows for each model correspond to results obtained before and after the application of stochastic

weight averaging (SWA), where the first epoch indicates the SWA start step, and the second corresponds to the final averaged model. The consistency of test accuracy before and after SWA highlights the stability and robustness of the fine-tuning procedure. Also, the key point is that for all U-MLIPs, with the same hyperparameters, we can all get comparable accuracy, showing that the robustness of MACE-based foundation models and its fine-tuning.

Figure 4 compares the test accuracy of models fine-tuned from different MACE foundation models with that of training-from-scratch, evaluated at different fractions of the training dataset (10%, 30%, 50%, 75% and 100%). Figure 4(a) shows the RMSE in energy, while Figure 4(b) shows the RMSE in forces. Across all dataset sizes, fine-tuned models (MP-0b3, MPA-0, OMAT-0) consistently achieve lower errors than training-from-scratch, particularly at lower data fractions, demonstrating the data efficiency and accuracy benefits of fine-tuning. The error bars remain small across experiments, indicating robustness against training noise. Overall, these results highlight the strength of MACE foundation models as effective starting points for fine-tuning, enabling improved performance even with limited training data.

#### B. Silicon

We use silicon studied in [108] as a representative system to evaluate the out-of-distribution generalization of fine-tuning, since the potential risk of overfitting has attracted considerable attention in the fine-tuning of foundation models. The training set consists of standard bulk and defect configurations, while the test set includes grain boundaries, di-interstitials, stacking fault paths, and an amorphous configuration. Unlike a random split, the test set contains configurations entirely distinct from the training data, providing a stringent benchmark for assessing extrapolative performance.

The previous example highlighted the data efficiency of fine-tuning with limited training data. In this case, we focus on ensemble-uncertainty-based active learning, where candidate configurations are filtered according to prediction uncertainty (cf. Section II B 2) following the procedure in [79]. This reduces the training set to only 5% of the original data. We then train a MACE model on this reduced dataset and compare its accuracy with that obtained from direct fine-tuning.

Figure 5 compares the performance of the foundation MACE model (MACE-MPA-0) with its fine-tuned counterpart (MACE-FT): (a) energy RMSE per atom (meV/atom), and (b) force RMSE (meV/Å). In both metrics, MACE-FT achieves substantially lower errors than MACE-MPA-0, indicating improved agreement with the reference data even for the out-of-distribution test set. This demonstrates that fine-tuning does not suffer from generalization issues in foundation models. A second conclusion is drawn from the comparison with

TABLE IV. Test accuracy of different fine-tuned MACE foundation models, reported as RMSE of energy (E) and force (F), relative to DFT values. “Relative F” denotes the force RMSE normalized by the standard deviation of DFT forces. Each model is evaluated before and after SWA, with the epoch number indicating the training step. Fine-tuning consistently achieves good accuracy regardless of which foundation model is used.

Model (epoch)	RMSE E (meV/atom)	RMSE F (meV/Å)	Relative F (%)
MACE-MP-0b3 (110)	0.32	16.15	1.92
MACE-MP-0b3 (148)	0.28	15.84	1.87
MACE-MPA-0 (110)	0.30	15.07	1.79
MACE-MPA-0 (146)	0.27	14.88	1.77
MACE-OMAT-0 (110)	0.35	15.46	1.84
MACE-OMAT-0 (149)	0.27	15.32	1.82

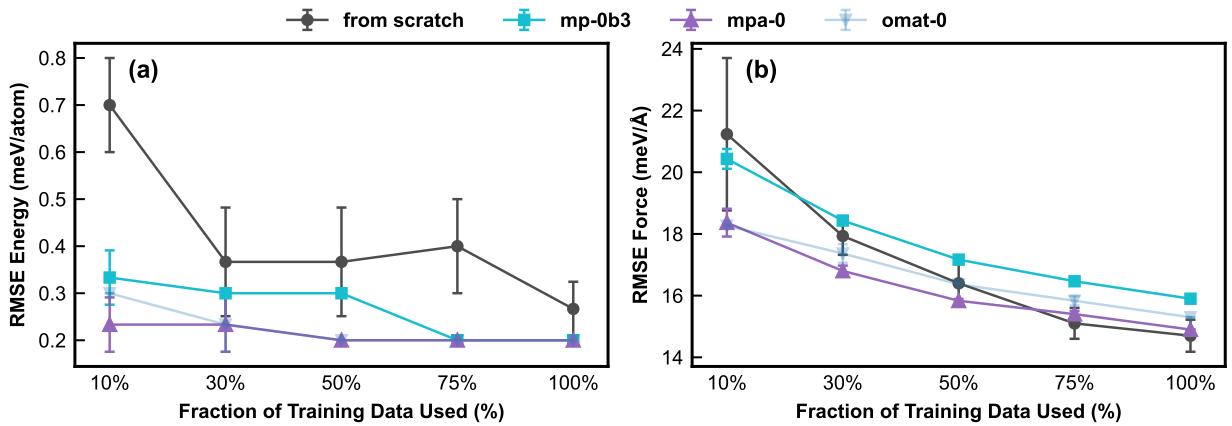


FIG. 4. Comparison between fine-tuning and training-from-scratch in terms of data efficiency: (a) RMSE in energy (meV/atom) and (b) RMSE in forces (meV/Å). Error bars represent one standard deviation, computed from three independent runs, with markers showing the corresponding mean values.

models trained on uncertainty-filtered datasets (labeled “MACE-AL” in Figure 5). Although the filtered set contains a similar number of samples, fine-tuning still yields superior accuracy. This highlights that fine-tuning not only ensures generalization but also achieves better data efficiency than training from scratch on a reduced dataset obtained through UQ-based active learning.

### C. Molybdenum

In this example, we evaluate the capability of fine-tuned U-MLIPs to accurately describe mechanical properties associated with crystalline defects in metallic systems. Molybdenum (Mo), a body-centered cubic (BCC) transition metal, is chosen as a representative case due to its well-characterized elastic properties and defect energetics, including vacancy formation and migration [109].

Crystalline defects play a key role in determining the plasticity, strength, and diffusion behavior of metals. However, accurately capturing their highly localized and stress-sensitive nature remains a challenge for many interatomic potentials. This example is therefore designed

to assess whether fine-tuning improves the quantitative prediction of defect-related energies and transition states in Mo. The atomic structure of bulk Mo is shown in Figure 6. The dataset used for fine-tuning is taken from Ref. [109].

*Bulk validation* — Table V compares the elastic constants ( $C_{11}$ ,  $C_{12}$ ,  $C_{44}$ ), bulk modulus ( $B$ ), and Poisson ratio ( $\nu$ ) of molybdenum predicted by different interatomic potentials (EAM, GAP, MACE-MP-0b3, and fine-tuned MACE-MP-0) against DFT and experimental reference values. The numbers in parentheses indicate the percentage error relative to experiment. The results show that fine-tuning the MACE-MP-0 model significantly improves its accuracy across all properties, reducing the error from 45.91% to 2.58% for  $C_{11}$ , from 56.88% to 14.77% for  $C_{44}$ , and from 48.27% to 3.45% for  $\nu$ . While the original MACE-MP-0b3 underestimates elastic constants and bulk modulus, fine-tuning closes the gap to both DFT and experimental values, achieving accuracy comparable to or better than EAM and GAP in most properties. These results highlight the effectiveness of fine-tuning in improving the mechanical property predictions of U-MLIPs.

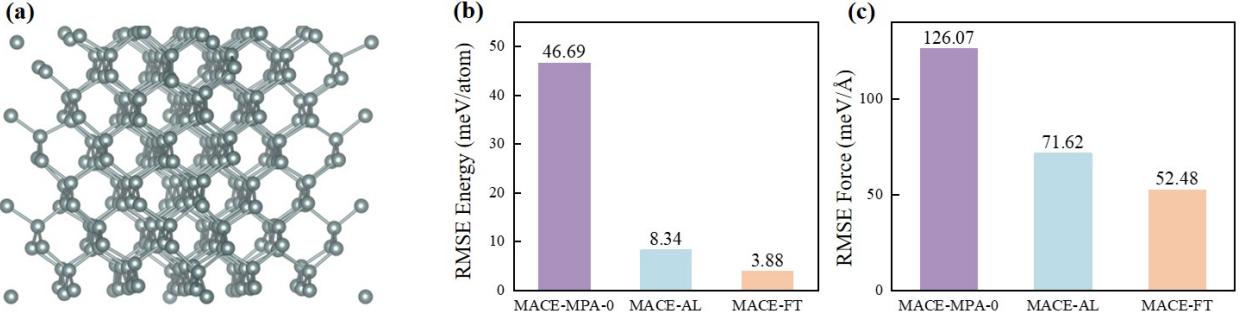


FIG. 5. Comparison of a foundation MACE model (MACE-MPA-0) and its fine-tuned counterpart (MACE-FT) on the silicon out-of-distribution test set. (a) Atomic structure; (b) energy RMSE per atom (meV/atom); (c) force RMSE (meV/Å). In both cases, MACE-FT achieves significantly lower errors than MACE-MPA-0, demonstrating improved accuracy and generalization. Moreover, MACE-FT also outperforms models trained from uncertainty-filtered datasets (MACE-AL).

TABLE V. Elastic constants  $C_{ij}$ , bulk modulus  $B$ , and Poisson ratio  $\nu$ , as obtained with the GAP, tabGAP, EAM/FS, and the NNIP potentials, compared to DFT calculations performed in this work and experimental data. Values in parentheses report the magnitude of the percentage error with respect to the experimental value.

	EAM [110]	GAP [111]	MACE-MP-0b3	Fine-tuning	DFT [109]	Experiment [112]
$C_{11}$ (GPa)	465 (0.22%)	478 (3.02%)	251 (45.91%)	452 (2.58%)	459	464
$C_{12}$ (GPa)	161 (1.26%)	166 (4.40%)	189 (18.87%)	174 (9.43%)	162	159
$C_{44}$ (GPa)	109 (0%)	108 (0.92%)	47 (56.88%)	82 (14.77%)	97	109
$B$ (GPa)	263 (5.20%)	270 (8.00%)	210 (16.00%)	267 (6.80%)	262	250
$\nu$	0.26 (10.34%)	0.26 (10.34%)	0.43 (48.27%)	0.28 (3.45%)	0.30	0.29

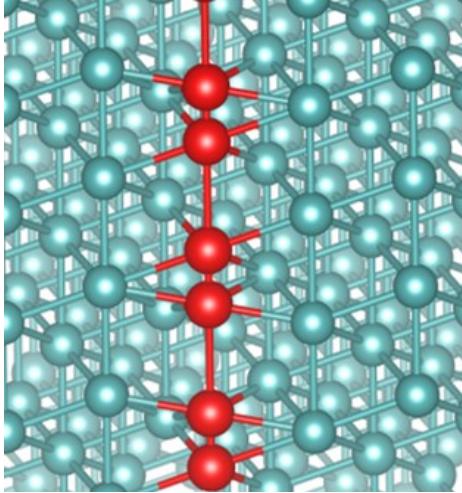


FIG. 6. Atomic structure of BCC Mo containing a dislocation and the associated local lattice distortion. The defect core is highlighted in red.

*Generalized stacking fault energy* — Figure 7 shows the generalized stacking fault energy (GSFE) profiles of Mo along the  $\langle 110 \rangle$  (panel (a)) and  $\langle 112 \rangle$  (panel (b)) directions. Table VI provides the corresponding GSFE barrier errors relative to DFT. The results compare the

MACE foundation model, two fine-tuned models (FT-Default, which employs the default loss weights, and FT, which uses an increased force loss weight of 100.0), and the DFT reference data. The foundation model clearly underestimates the GSFE in both directions, leading to lower energy barriers. After fine-tuning, the predictions become much closer to the DFT values, especially near the peak and along the entire fault path. Between the two fine-tuned models, FT performs slightly better than FT-Default, suggesting that adjusting the force weighting can further improve accuracy.

#### D. Graphene–Water Interface

Graphene–water interfaces are prototypical solid–liquid systems with wide-ranging applications in sensing, electrochemistry, and nanofluidics. Capturing the interfacial structure and dynamics of such systems presents a significant challenge for interatomic potentials, especially in describing long-range interactions and subtle surface effects. This example illustrates the capability of fine-tuned U-MLIPs to model solid–liquid interfaces accurately, using a representative graphene–water system. The atomic structure of the graphene–water interface is illustrated in Figure 8. The system contained over 372 atoms and was simulated for

TABLE VI. GSFE barrier errors relative to DFT are reported for the  $\langle 110 \rangle$  and  $\langle 121 \rangle$  slip directions, comparing three interatomic potential models: Fine-Tuning, FT-Default, and MACE-MP-0b3.

Slip Directions	Fine-Tuning	FT-Default	MACE-MP-0b3
$\langle 110 \rangle$	<b>0.17</b>	0.23	0.88
$\langle 121 \rangle$	<b>0.21</b>	0.36	0.78

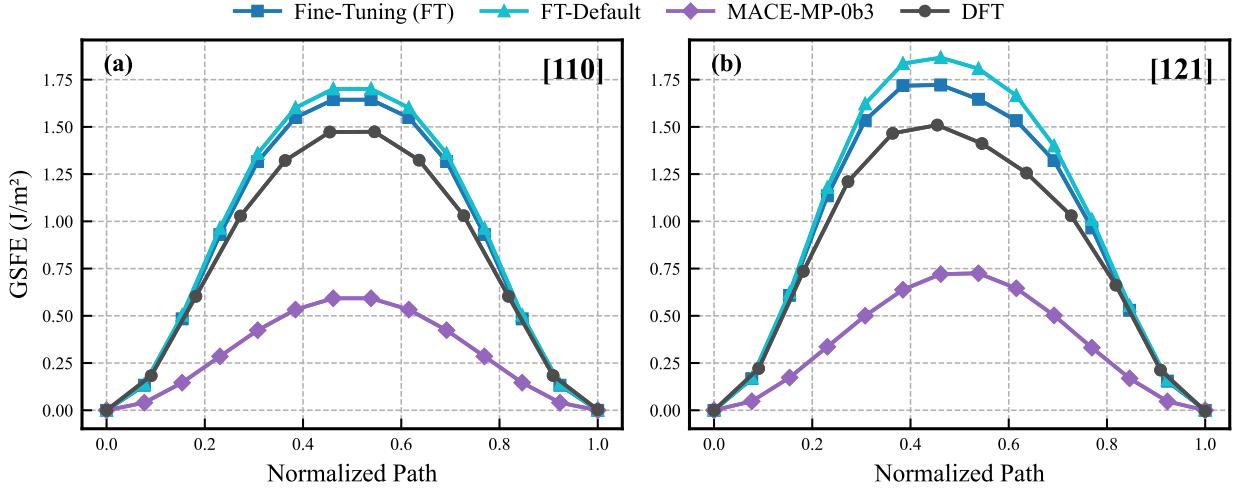


FIG. 7. GSFE profiles of Mo along the  $\langle 110 \rangle$  (panel (a)) and  $\langle 112 \rangle$  (panel (b)) directions. Two fine-tuned models are compared: FT-Default, which uses the default loss weights, and FT, which employs a force loss weight of 100.0.

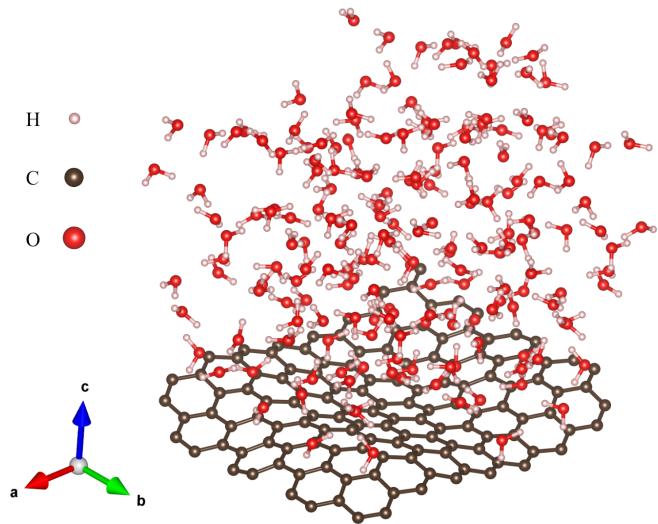


FIG. 8. Atomic structure of the graphene–water interface used in this study.

200 ps with a 1 fs timestep.

The graphene–water dataset used to fine-tune the U-MLIP was generated by pre-equilibrating a periodic graphene–water interface with classical molecular dynamics (LAMMPS) [113, 114]. Graphene was modeled with a Tersoff bond-order potential [115, 116],

and water with a rigid three-site model constrained by SHAKE; short-range interactions were described by Lennard–Jones terms [117], and long-range electrostatics by Ewald summation under periodic boundary conditions. Canonical (NVT) sampling furnished equilibrated configurations, from which temporally uncorrelated snapshots were randomly selected for ab initio labeling. Single-point DFT labels were obtained in VASP [118] using the PAW [119] formalism and the PBE exchange–correlation functional [120] augmented with D3 dispersion [121] to capture graphene–water physisorption. cutoff of 400 eV was chosen for the plane-wave basis, and electronic self-consistency was achieved when the total-energy change fell below  $1 \times 10^{-6}$  eV. Geometry optimisations were considered converged once the residual forces on all atoms were smaller than  $0.02 \text{ eV } \text{\AA}^{-1}$ .

Figure 9 compares molecular dynamics simulations of a graphene–water interface performed using LAMMPS–MACE with the fine-tuned model (MACE-FT-MD) and ab initio molecular dynamics (AIMD). Figure 9(a) shows the temperature evolution, with both simulations maintaining comparable temperature fluctuations around 300 K. Figure 9(b) presents the oxygen number-density profile along the  $z$ -axis. The first peak corresponds to the graphene–water interfacial distance,  $dgw = 0.35 \text{ nm}$ , which is in good agreement with the experimental value of 0.36 nm [122]. Due to the hydrophobic nature of graphene, a depletion layer forms at the interface, and

TABLE VII. Computational cost (wall-clock time in seconds) for 10000-step molecular dynamics simulations with and without the use of the `cuEquivalence` kernel. The first column indicates the total number of atoms in the system, while the remaining columns report the elapsed time for each configuration. “With” denotes simulations utilizing the `cuEquivalence` kernel, and “Without” corresponds to runs without it.

Atoms	128	372	1038	2900	Scaling
With (s)	710	800	1126	2619	<b>0.41</b>
Without (s)	658	1008	2031	5084	0.76

$\text{dgw}$  is defined to quantify this interfacial gap. This distance is a critical physical quantity for evaluating the accuracy of the interaction potentials and demonstrates the effectiveness of the fine-tuned model.

Table VII reports the computational cost (wall-clock time in seconds) for 10,000-step molecular dynamics simulations performed with and without the `cuEquivalence` kernel, across systems of varying sizes (128 to 2900 atoms). For each configuration, simulations using `cuEquivalence` (“With”) show consistently lower elapsed times compared to runs without it (“Without”), especially for larger systems. For instance, at 2900 atoms, `cuEquivalence` reduces the runtime from 5084 seconds to 2619 seconds. The scaling factor improves from 0.76 (without `cuEquivalence`) to 0.41 (with `cuEquivalence`), indicating better parallel efficiency with the kernel enabled. These results suggest that the new kernel can serve as an efficient building block for large-scale and long-timescale molecular dynamics simulations.

### E. Molecule/Oxide Interface

As a representative case of molecule–oxide interactions, we consider a gold dimer ( $\text{Au}_2$ ) adsorbed on the  $\text{MgO}(001)$  surface with aluminum doping, following [123]. The dataset is also from the authors of [123]. This system has been widely employed to benchmark long-range effects such as charge transfer and polarization at oxide surfaces. It thus provides a stringent test of whether fine-tuned foundation models can accurately capture long-range physics beyond the reach of conventional MLIPs.

Figure 10 compares the performance of the foundation MACE model (MACE-MPA-0) with its fine-tuned counterpart (MACE-FT): (a) energy RMSE per atom (meV/atom), and (b) force RMSE (meV/ $\text{\AA}$ ). In both metrics, MACE-FT achieves substantially lower errors than MACE-MPA-0, demonstrating improved agreement with the reference data (lower is better). The relatively large error of MACE-MPA-0 arises because its reported value corresponds to the step-0 output during training, reflecting a mismatch between the DFT settings used in this dataset and those employed in the original

MPA training. Notably, even without explicitly modeling long-range interactions, the fine-tuned foundation model outperforms recent state-of-the-art methods reported in [123]. The CACE-LR model, which incorporates a latent Ewald summation to capture long-range effects, already delivers strong accuracy; yet our fine-tuned model achieves comparable or even superior performance. This suggests that fine-tuning can implicitly adapt to capture aspects of long-range physics, highlighting the potential of foundation models as a flexible and powerful basis for tackling complex interfacial systems.

### F. Solid–Solid Interface

Atomistic modeling of solid–solid interfaces is essential for understanding the synthesizability and stability of materials [124]. Such interfaces are notoriously challenging to model because their heterogeneous nature often requires long-period supercells and an accurate treatment of long-range interactions, including charge transfer and polarization, which are typically beyond the scope of conventional MLIPs. To assess whether fine-tuned foundation models can capture these effects, we consider a representative heterointerface from [123], namely  $\text{LiCl}(001)/\text{GaF}_3(001)$ . The atomic structure is shown in Figure 11.

Table VIII reports the test RMSEs for energies and forces before and after fine-tuning, together with reference results from [123]. The CACE-LR method, which employs a latent Ewald summation to explicitly describe long-range interactions, already achieves impressive accuracy. Remarkably, however, the fine-tuned foundation model (MACE-FT) performs even better, despite lacking any explicit long-range correction. We attribute this to the fact that large pre-trained models may implicitly encode aspects of long-range physics during pre-training, which fine-tuning then leverages and refines for the target interface. This observation highlights an intriguing direction: foundation models, when fine-tuned, might generalize long-range effects in ways not captured by traditional MLIPs. We plan to further investigate this phenomenon in future work.

TABLE VIII. Test RMSEs for energies ( $E$ , meV/atom) and forces ( $F$ , meV/ $\text{\AA}$ ) of the  $\text{LiCl}(001)/\text{GaF}_3(001)$  interface. Results for CACE-LR are taken from [123]; “—” indicates values not reported. The fine-tuned MACE-MPA-0 (MACE-FT) achieves the best accuracy across both metrics.

	MACE-MPA-0	CACE-LR	MACE-FT
$E$ (meV/atom)	97.04	—	<b>0.09</b>
$F$ (meV/ $\text{\AA}$ )	179.2	67.8	<b>18.23</b>

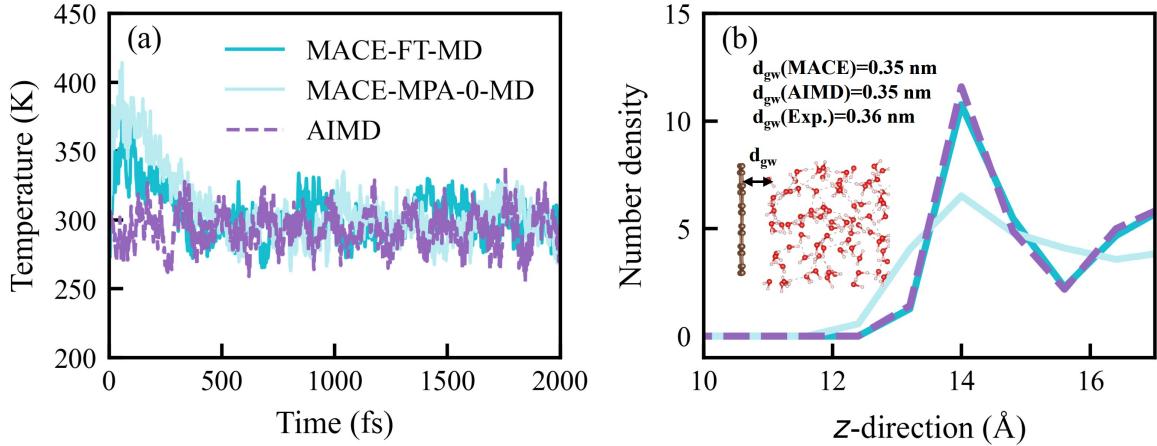


FIG. 9. Comparison of graphene–water interface simulations using MACE-FT-MD, MACE-MPA-0-MD and AIMD. (a) Temperature evolution over time. (b) Oxygen number-density profile along the z-axis; The first peak corresponds to the graphene–water distance,  $d_{gw} = 0.35 \text{ nm}$ , which is in good agreement with the experimental value of  $0.36 \text{ nm}$  [122]. The inset illustrates the definition of  $d_{gw}$ . To make a comparison, we also plot the MD trajectory for MACE-MPA-0 model without fine-tuning.

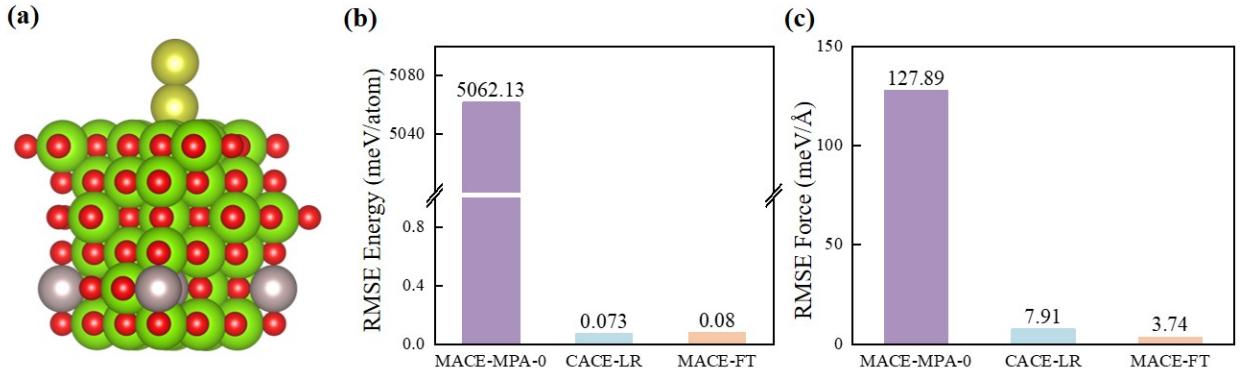


FIG. 10. Performance comparison between a foundation MACE model (MACE-MPA-0) and its fine-tuned counterpart (MACE-FT) on the doped Au-MgO surface. (a) Atomic structure; (b) energy RMSE per atom (meV/atom); (c) force RMSE (meV/Å). MACE-FT achieves consistently lower errors than MACE-MPA-0 and reaches accuracy comparable to CACE-LR [123], a state-of-the-art method with explicit long-range interactions.

#### IV. SUMMARY AND PERSPECTIVE

This tutorial provides a practical guide to fine-tuning universal machine-learned interatomic potentials (U-MLIPs), using the MACE-MP-0 model as a representative example. We cover the end-to-end process—from dataset preparation and hyperparameter selection to model training and application—with the goal of helping researchers efficiently adapt pre-trained models to their systems of interest. The effectiveness of fine-tuning is demonstrated through representative examples, including solid-state electrolytes, stacking fault defects in metals, surface interactions in low-dimensional materials, and more complicated heterointerfaces. To support reproducibility, all code and data are made publicly available, along with scripts for running molecular dynamics

simulations using the RBMD platform.

While this tutorial focuses on MACE-MP-0, the strategies discussed are readily transferable to other U-MLIP frameworks. We hope that this work will help lower the barrier to entry for applying fine-tuned MLIPs in atomistic simulations, and encourage their broader use in both academic and industrial settings.

That said, caution is required when fine-tuning or applying MLIPs. These models approximate the potential energy surface purely from data and do not inherently incorporate physical laws or constraints. Their accuracy is ultimately determined by the coverage and quality of the training dataset. We point out several important considerations and open directions for future development of MLIPs.

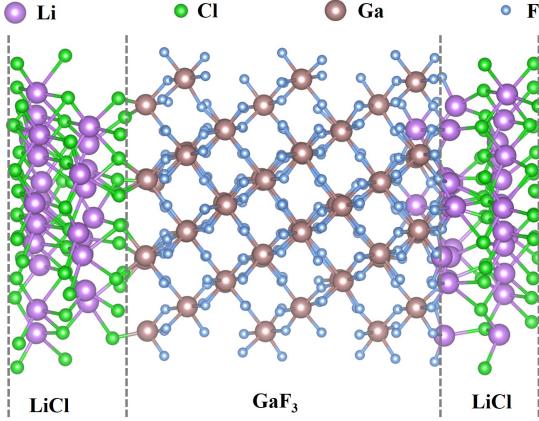


FIG. 11. Atomic structure of the LiCl(001)/GaF<sub>3</sub>(001) interface used in this study.

*a. Uncertainty Quantification and Active Fine-Tuning.* Uncertainty quantification (UQ) is essential for evaluating model reliability and guiding data selection in active learning and fine-tuning. Techniques such as ensemble variance, dropout inference, and Bayesian approximations help identify regions of high predictive uncertainty, enabling targeted sampling and noise reduction. In fine-tuning, UQ reduces data redundancy and ensures that added configurations meaningfully enhance model performance. A central challenge is the automated construction of fine-tuning datasets informed by uncertainty, task relevance, or physical constraints. Once such datasets can be systematically generated, selecting or adapting models becomes significantly more efficient. Developing robust pipelines for uncertainty-aware data generation is thus key to scalable, task-adaptive fine-tuning [125].

*b. Building Intermediate-Sized MLIPs for Specific Tasks.* Although U-MLIPs demonstrate excellent generalization across a wide range of materials systems, they may not be optimal for small- to medium-sized systems or domain-specific tasks. In such cases, it may be more effective to either fine-tune a compact pre-trained model or train a lightweight architecture from scratch on a focused dataset [126, 127]. This approach offers a favorable trade-off between interpretability, computational efficiency, and predictive accuracy—particularly when computational resources are limited or when real-time inference is required for large-scale simulations. Lightweight variants of ACE and other descriptor-based models remain valuable tools for such targeted applications.

*c. The Need for Comprehensive Benchmarking Platforms for MLIPs.* As MLIPs continue to evolve into foundation models with broad applicability, there is an increasing need for benchmarking platforms that go beyond traditional accuracy metrics such as RMSE. Evaluating an MLIP's true usefulness requires not only numerical accuracy, but also physical fidelity, generalization

ability, and reliability in practical simulations. Conventional benchmarks often fail to capture these dimensions, making it difficult to assess whether a model is suitable for deployment in real-world applications. To address this gap, recent efforts have begun to develop more comprehensive evaluation frameworks that emphasize model robustness, transferability, and physics-informed behavior. One notable example is *MLIP Arena* [128], a newly proposed platform that offers architecture-agnostic, task-oriented benchmarking of open-source, open-weight MLIPs. It represents a promising step toward standardized, holistic assessment of model performance in complex materials modeling tasks.

*d. Post-Training Strategies.* After pre-training a foundation model, post-training serves as a crucial step to further adapt and optimize the model for specific downstream applications. One effective post-training technique is *model distillation*, in which a simpler model (the “student”) is trained to replicate the behavior and predictive performance of a larger, more complex model (the “teacher”—in this case, U-MLIPs). This is typically achieved by using the teacher’s outputs or internal representations as supervisory signals for training the student [129]. Recent work [130] demonstrates that matching the Hessians of energy predictions between the teacher and student enables effective distillation. In particular, large-scale foundation models can be distilled into compact, task-specific MLIPs tailored to a particular chemical subdomain, achieving up to 50× speedup in inference time compared to the original foundation model. Future investigations on this topic will be conducted using the RBMD platform.

## V. ACKNOWLEDGEMENTS

We gratefully acknowledge the original developers of MACE, particularly Prof. Gábor Csányi’s group, for their insightful discussions and valuable contributions. We also appreciate the helpful suggestions and exchanges on the MACE GitHub repository (<https://github.com/ACEsuit/mace>) that have informed and improved this tutorial on fine-tuning MACE foundation models. Readers are encouraged to consult the official MACE documentation (<https://mace-docs.readthedocs.io/en/latest/>), upon which parts of this tutorial are based. Finally, we gratefully acknowledge the author of the GitHub repository (<https://github.com/BingqingCheng/cace-lr-fit>) for generously sharing the dataset that enabled our last two numerical examples.

## VI. CODES AND DATA AVAILABILITY

The datasets and source code supporting all numerical results presented in this work are publicly available in the tutorial repository: <https://github.com/>

`John2021-hub/mace-ft-tutorial.git`.

## VII. MACE ARCHITECTURE

In this subsection, we introduce the main idea behind the construction of the MACE architecture. We provide only a brief overview here and refer interested readers to [6] for more detailed constructional insights.

First, a graph is defined by connecting two nodes (atoms) with an edge if they are within each other's local environment. The local environment,  $\mathcal{N}(i)$ , consists of all atoms  $j$  surrounding a central atom  $i$  such that  $\|\mathbf{r}_{ij}\| \leq r_{\text{cut}}$ , where  $\mathbf{r}_{ij}$  is the vector from atom  $i$  to atom  $j$ , and  $r_{\text{cut}}$  is a cutoff hyperparameter. The feature vector of node  $i$  is denoted by  $\mathbf{h}_i^{(t)}$ , expressed in the spherical harmonic basis, with indices  $l$  and  $m$ . The superscript  $t$  represents the iteration step (analogous to the "layers" in graph neural networks' message-passing). These node features,  $\mathbf{h}_i^{(t)}$ , reflect the chemical environment of the atoms.

It is worth noting that all MACE models consist of only two layers. Due to the specific construction, MACE can be regarded as a higher-order message-passing neural network. This enables it to achieve comparable expressiveness with just two layers, which is a key strength of the MACE model.

The node features on 0-th layer denoted as  $\mathbf{h}_i^{(0)}$  are initialized as a (learnable) embedding of the chemical elements with atomic numbers  $z_i$  into  $k$  channels:

$$\mathbf{h}_{i,k00}^{(0)} = \sum_z W_{kz} \delta_{zz}. \quad (1)$$

This type of mapping has been widely applied to graph neural networks [7, 131] and other models [38, 132].

Next, for each atom, the features of its neighbors are combined with the interatomic displacement vectors,  $\mathbf{r}_{ij}$ , to form the one-particle basis  $\phi_{ij,knl_1l_2m_2}^{(t)}$ . The construction of MACE employs the idea from the ACE architecture, which is a natural many-body representation for capturing the symmetry of many-body particle systems. The radial distance  $r_{ij}$  is used as an input into a learnable radial function  $R(r_{ij})$ , with several outputs that correspond to different ways in which the displacement vector and node features can be combined while maintaining equivariance [133]:

$$\begin{aligned} \phi_{ij,knl_1l_2m_3}^{(t)} &= \sum_{l_1 m_1 l_2 m_2} C_{\eta_1, l_1 m_1, l_2 m_2}^{l_3 m_3} R_{knl_1l_2l_3}^{(t)}(r_{ij}) \\ &\quad Y_{l_1 m_1}^{m_1}(\hat{r}_{ij}) h_{j,kl_2m_2}^{(t)}, \end{aligned} \quad (2)$$

where  $Y_l^m$  are the spherical harmonics, and  $C_{\eta_1, l_1 m_1, l_2 m_2}^{l_3 m_3}$  denotes the Clebsch-Gordan coefficients. There are multiple ways of constructing an equivariant combination

with a given symmetry  $(l_3, m_3)$ , and these multiplicities are enumerated by the path index  $\eta_1$  [37].

The one-particle basis  $\phi$  is summed over the neighborhood and expanded as a linear combination of  $k$  channels with learnable weights to form the permutation-invariant atomic basis  $A_i$ :

$$A_{i,kl3m3}^{(t)} = \sum_{\tilde{k}, \eta_1} W_{\tilde{k}kk\eta_1l_3} \sum_{j \in \mathcal{N}(i)} \phi_{ij,\tilde{k}\eta_1l_3m_3}^{(t)}. \quad (3)$$

Higher-order (many-body) symmetric features are created on each atom by taking products of the atomic basis,  $A$ , with itself  $\nu$  times, resulting in the "product basis". The product basis is then contracted with the generalised Clebsch-Gordan coefficients  $C_{\eta_\nu lm}^{LM}$  to obtain the equivariant higher-order basis,  $B_i$  [37]:

$$B_{i,\eta_\nu kLM}^{(t),\nu} = \sum_{lm} C_{\eta_\nu lm}^{LM} \prod_{\xi=1}^{\nu} A_{i,kl\xi m_\xi}^{(t)}, \quad (4)$$

where bold  $lm$  denotes the  $\nu$ -tuple of  $l$  and  $m$  values and similarly to Equation (2),  $\eta_\nu$  enumerates the number of possible couplings to create the features with equivariance  $LM$ .

Finally, a "message"  $m_i$  is created on each atom as a learnable linear combination of the equivariant many-body features:

$$m_{i,kLM}^{(t)} = \sum_{\nu} \sum_{\eta_\nu} W_{z i \eta_\nu kLM}^{(t),\nu} B_{i,\eta_\nu kLM}^{(t),\nu}. \quad (5)$$

The recursive update of the node features ( $t : 0 \rightarrow 2$ ) is obtained by adding the message to the atoms' features from the previous iteration, with weights that depend on the chemical element ( $z_i$ ) that are also responsible for the mixing of the chemical embedding  $k$  channels:

$$h_{i,kLM}^{(t+1)} = \sum_{\tilde{k}} W_{kL,\tilde{k}}^{(t)} m_{i,\tilde{k}LM}^{(t)} + \sum_{\tilde{k}} W_{kziL,\tilde{k}}^{(t)} h_{i,\tilde{k}LM}^{(t)}. \quad (6)$$

The site energy is a sum of read-out functions applied to node features from the first and second layers. The read-out function is defined as a linear combination of rotationally invariant node features for the first layer, and as a multi-layer perceptron (MLP) for the second layer.

$$E_i = \sum_{t=1}^2 E_i^{(t)} = \sum_{t=1}^2 \mathcal{R}^{(t)} \left( h_i^{(t)} \right), \quad (7)$$

where

$$\mathcal{R}^{(t)} \left( h_i^{(t)} \right) = \begin{cases} \sum_k W_k^{(t)} h_{i,k00}^{(t)} & \text{for } t = 1 \\ \text{MLP} \left( \left\{ h_{i,k00}^{(t)} \right\}_k \right) & \text{for } t = 2 \end{cases}. \quad (8)$$

The forces and stresses on the atoms are calculated by taking analytical derivatives of the total potential energy with respect to the positions of the atoms.

- [1] J. Behler and M. Parrinello, Generalized neural-network representation of high-dimensional potential-energy surfaces, *Phys. Rev. Lett.* **98**, 146401 (2007).
- [2] A. P. Bartók, M. C. Payne, R. Kondor, and G. Csányi, Gaussian approximation potentials: The accuracy of quantum mechanics, without the electrons, *Phys. Rev. Lett.* **104**, 136403 (2010).
- [3] A. V. Shapeev, Moment tensor potentials: A class of systematically improvable interatomic potentials, *Multiscale Model. Simul.* **14**, 1153 (2016).
- [4] H. Wang, L. Zhang, J. Han, *et al.*, DeePMD-kit: A deep learning package for many-body potential energy representation and molecular dynamics, *Comput. Phys. Commun.* **228**, 178 (2018).
- [5] R. Drautz, Atomic cluster expansion for accurate and transferable interatomic potentials, *Phys. Rev. B* **99**, 014104 (2019).
- [6] I. Batatia, D. P. Kovacs, G. Simm, C. Ortner, and G. Csányi, MACE: Higher order equivariant message passing neural networks for fast and accurate force fields, *Adv. Neural Inf. Process. Syst.* **35** (2022).
- [7] K. Schütt, P.-J. Kindermans, H. E. Sauceda Felix, S. Chmiela, A. Tkatchenko, and K.-R. Müller, Schnet: A continuous-filter convolutional neural network for modeling quantum interactions, *Adv. Neural Inf. Process. Syst.* **30**, 992 (2017).
- [8] S. Batzner, A. Musaelian, L. Sun, M. Geiger, J. P. Mailoa, M. Kornbluth, N. Molinari, T. E. Smidt, and B. Kozinsky, E (3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials, *Nat. Commun.* **13**, 2453 (2022).
- [9] B. Cheng, Cartesian atomic cluster expansion for machine learning interatomic potentials, *npj Comput. Mater.* **10**, 157 (2024).
- [10] A. Musaelian, S. Batzner, A. Johansson, L. Sun, C. J. Owen, M. Kornbluth, and B. Kozinsky, Learning local equivariant representations for large-scale atomistic dynamics, *Nat. Commun.* **14**, 579 (2023).
- [11] J. S. Smith, O. Isayev, and A. E. Roitberg, ANI-1: an extensible neural network potential with DFT accuracy at force field computational cost, *Chem. Sci.* **8**, 3192 (2017).
- [12] A. Bochkarev, Y. Lysogorskiy, and R. Drautz, Graph atomic cluster expansion for semilocal interactions beyond equivariant message passing, *Phys. Rev. X* **14**, 021036 (2024).
- [13] A. P. Thompson, L. P. Swiler, C. R. Trott, S. M. Foiles, and G. J. Tucker, Spectral neighbor analysis method for automated generation of quantum-accurate interatomic potentials, *J. Comput. Phys.* **285**, 316 (2015).
- [14] S. R. Xie, M. Rupp, and R. G. Hennig, Ultra-fast interpretable machine-learning potentials, *npj Comput. Mater.* **9**, 162 (2023).
- [15] A. H. Larsen, J. J. Mortensen, J. Blomqvist, I. E. Castelli, R. Christensen, M. Dułak, J. Friis, M. N. Groves, B. Hammer, C. Hargus, *et al.*, The atomic simulation environment—a python library for working with atoms, *J. Phys.: Condens. Matter* **29**, 273002 (2017).
- [16] 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.*, LAMMPS-a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales, *Comput. Phys. Commun.* **271**, 108171 (2022).
- [17] O. T. Unke, S. Chmiela, H. E. Sauceda, M. Gastegger, I. Poltavsky, K. T. Schutt, A. Tkatchenko, and K.-R. Müller, Machine learning force fields, *Chem. Rev.* **121**, 10142 (2021).
- [18] V. Botu, R. Batra, J. Chapman, and R. Ramprasad, Machine learning force fields: construction, validation, and outlook, *J. Phys. Chem. C* **121**, 511 (2017).
- [19] R. Jacobs, D. Morgan, S. Attarian, J. Meng, C. Shen, Z. Wu, C. Y. Xie, J. H. Yang, N. Artrith, B. Blaiszik, *et al.*, A practical guide to machine learning interatomic potentials—status and future, *Curr. Opin. Solid State Mater. Sci.* **35**, 101214 (2025).
- [20] F. Musil, A. Grisafi, A. P. Bartók, C. Ortner, G. Csányi, and M. Ceriotti, Physics-inspired structural representations for molecules and materials, *Chem. Rev.* **121**, 9759 (2021).
- [21] I. Poltavsky and A. Tkatchenko, Machine learning force fields: Recent advances and remaining challenges, *J. Phys. Chem. Lett.* **12**, 6551 (2021).
- [22] 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.*, A foundation model for atomistic materials chemistry, arXiv preprint arXiv:2401.00096 (2023).
- [23] B. Deng, P. Zhong, K. Jun, J. Riebesell, K. Han, C. J. Bartel, and G. Ceder, Chgnet as a pretrained universal neural network potential for charge-informed atomistic modelling, *Nat. Mach. Intell.* **5**, 1031 (2023).
- [24] A. Merchant, S. Batzner, S. S. Schoenholz, M. Aykol, G. Cheon, and E. D. Cubuk, Scaling deep learning for materials discovery, *Nature* **624**, 80 (2023).
- [25] D. Zhang, X. Liu, X. Zhang, C. Zhang, C. Cai, H. Bi, Y. Du, X. Qin, A. Peng, J. Huang, *et al.*, DPA-2: a large atomic model as a multi-task learner, *npj Comput. Mater.* **10**, 293 (2024).
- [26] K. Choudhary, B. DeCost, L. Major, K. Butler, J. Thiyyagalingam, and F. Tavazza, Unified graph neural network force-field for the periodic table: solid state applications, *Digit. Discov.* **2**, 346 (2023).
- [27] C. Chen and S. P. Ong, A universal graph deep learning interatomic potential for the periodic table, *Nat. Comput. Sci.* **2**, 718 (2022).
- [28] M. Awais, M. Naseer, S. Khan, R. M. Anwer, H. Cholakkal, M. Shah, M.-H. Yang, and F. S. Khan, Foundation models defining a new era in vision: a survey and outlook, *IEEE Trans. Pattern Anal. Mach. Intell.* **47**, 2245 (2025).
- [29] L. Chanussot, A. Das, S. Goyal, T. Lavril, M. Shuaibi, M. Riviere, K. Tran, J. Heras-Domingo, C. Ho, W. Hu, *et al.*, Open catalyst 2020 (OC20) dataset and community challenges, *ACS Catal.* **11**, 6059 (2021).
- [30] J. M. Bowman, C. Qu, R. Conte, A. Nandi, P. L. Houston, and Q. Yu, The MD17 datasets from the perspective of datasets for gas-phase “small” molecule potentials, *J. Chem. Phys.* **156** (2022).
- [31] L. Barroso-Luque, M. Shuaibi, X. Fu, B. M. Wood, M. Dzamba, M. Gao, A. Rizvi, C. L. Zitnick, and Z. W. Ulissi, Open materials 2024 (omat24) inorganic materi-

- als dataset and models, arXiv preprint arXiv:2410.12771 (2024).
- [32] K. T. Schütt, H. E. Sauceda, P.-J. Kindermans, A. Tkatchenko, and K.-R. Müller, Schnet—a deep learning architecture for molecules and materials, *J. Chem. Phys.* **148**, 241722 (2018).
- [33] S. Takamoto, D. Okanohara, Q.-J. Li, and J. Li, Towards universal neural network interatomic potential, *J. Materiomics* **9**, 447 (2023).
- [34] Y. Park, J. Kim, S. Hwang, and S. Han, Scalable parallel algorithm for graph neural network interatomic potentials in molecular dynamics simulations, *J. Chem. Theory Comput.* **20**, 4857 (2024).
- [35] H. Yang, C. Hu, Y. Zhou, X. Liu, Y. Shi, J. Li, G. Li, Z. Chen, S. Chen, C. Zeni, *et al.*, Mattersim: A deep learning atomistic model across elements, temperatures and pressures, arXiv preprint arXiv:2405.04967 (2024).
- [36] Y.-L. Liao, B. Wood, A. Das, and T. Smidt, Equiformerv2: Improved equivariant transformer for scaling to higher-degree representations, ICLR (2024).
- [37] G. Dusson, M. Bachmayr, G. Csányi, R. Drautz, S. Ettner, C. van der Oord, and C. Ortner, Atomic cluster expansion: Completeness, efficiency and stability, *J. Comput. Phys.* **454**, 110946 (2022).
- [38] J. P. Darby, D. P. Kovács, I. Batatia, M. A. Caro, G. L. Hart, C. Ortner, and G. Csányi, Tensor-reduced atomic density representations, *Phys. Rev. Lett.* **131**, 028001 (2023).
- [39] Z. Li, T. Wen, Y. Zhang, X. Liu, C. Zhang, A. Pattamatta, X. Gong, B. Ye, H. Wang, L. Zhang, *et al.*, An extendable cloud-native alloy property explorer, arXiv preprint arXiv:2404.17330 (2024).
- [40] B. Focassio, L. P. M. Freitas, and G. R. Schleder, Performance assessment of universal machine learning interatomic potentials: Challenges and directions for materials' surfaces, *ACS Appl. Mater. Interfaces* **17**, 13111 (2024).
- [41] B. Deng, Y. Choi, P. Zhong, J. Riebesell, S. Anand, Z. Li, K. Jun, K. A. Persson, and G. Ceder, Systematic softening in universal machine learning interatomic potentials, *npj Comput. Mater.* **11**, 1 (2025).
- [42] H. Yu, M. Giantomassi, G. Materzanini, J. Wang, and G.-M. Rignanese, Systematic assessment of various universal machine-learning interatomic potentials, *Mater. Genome Eng. Adv.* **2**, e58 (2024).
- [43] E. O. Pyzer-Knapp, M. Manica, P. Staar, L. Morin, P. Ruch, T. Laino, J. R. Smith, and A. Curioni, Foundation models for materials discovery—current state and future directions, *npj Comput. Mater.* **11**, 61 (2025).
- [44] F. Shuang, Z. Wei, K. Liu, W. Gao, and P. Dey, Universal machine learning interatomic potentials poised to supplant DFT in modeling general defects in metals and random alloys, arXiv preprint arXiv:2502.03578 (2025).
- [45] H. Du, J. Hui, L. Zhang, and H. Wang, Universal machine learning interatomic potentials are ready for solid ion conductors, arXiv preprint arXiv:2502.09970 (2025).
- [46] H. Lee, V. I. Hegde, C. Wolverton, and Y. Xia, Accelerating high-throughput phonon calculations via machine learning universal potentials, *Mater. Today Phys.* **53**, 101688 (2025).
- [47] S. P. Niblett, P. Kourtis, I.-B. Magdău, C. P. Grey, and G. Csányi, Transferability of datasets between machine-learning interaction potentials, arXiv preprint arXiv:2409.05590 (2024).
- [48] L. Casillas-Trujillo, A. S. Parackal, R. Armiento, and B. Alling, Evaluating and improving the predictive accuracy of mixing enthalpies and volumes in disordered alloys from universal pretrained machine learning potentials, *Phys. Rev. Mater.* **8**, 113803 (2024).
- [49] W. Gao, T. Zhao, Y. Guo, J. Liang, H. Liu, M. Luo, Z. Luo, W. Qin, Y. Wang, Q. Zhou, *et al.*, RBMD: A molecular dynamics package enabling to simulate 10 million all-atom particles in a single graphics processing unit, *Commun. Comput. Phys.* (2025), to appear; preprint available at arXiv:2407.09315.
- [50] S. Jin, L. Li, and J.-G. Liu, Random batch methods (RBM) for interacting particle systems, *J. Comput. Phys.* **400**, 108877 (2020).
- [51] S. Jin, L. Li, Z. Xu, and Y. Zhao, A random batch ewald method for particle systems with coulomb interactions, *SIAM J. Sci. Comput.* **43**, B937 (2021).
- [52] J. Liang, P. Tan, Y. Zhao, L. Li, S. Jin, L. Hong, and Z. Xu, Superscalability of the random batch ewald method, *J. Chem. Phys.* **156**, 014114 (2022).
- [53] J. Liang, Z. Xu, and Y. Zhao, Improved random batch ewald method in molecular dynamics simulations, *J. Phys. Chem. A* **126**, 3583 (2022).
- [54] W. C. Witt, C. van der Oord, E. Gelžinytė, T. Järvinen, A. Ross, J. P. Darby, C. H. Ho, W. J. Baldwin, M. Sachs, J. Kermode, *et al.*, ACEpotentials.jl: A julia implementation of the atomic cluster expansion, *J. Chem. Phys.* **159**, 164101 (2023).
- [55] L. Zhang, B. Onat, G. Dusson, A. McSloy, G. Anand, R. J. Maurer, C. Ortner, and J. R. Kermode, Equivariant analytical mapping of first principles Hamiltonians to accurate and transferable materials models, *npj Comput. Mater.* **8**, 158 (2022).
- [56] Y. Wang, S. Patel, and C. Ortner, A theoretical case study of the generalization of machine-learned potentials, *Comput. Methods Appl. Mech. Eng.* **422**, 116831 (2024).
- [57] T. Torabi, C. Ortner, and Y. Wang, Surrogate models for vibrational entropy based on a spatial decomposition, *Multiscale Model. Simul.* **23**, 514 (2025).
- [58] Y. Wang, G. Csanyi, and C. Ortner, Many-body coarse-grained molecular dynamics with the atomic cluster expansion, arXiv preprint arXiv:2502.04661 (2025).
- [59] C. Vignac, A. Loukas, and P. Frossard, Building powerful and equivariant graph neural networks with structural message-passing, *Adv. Neural Inf. Process. Syst.* **33** (2020).
- [60] H. Maron, H. Ben-Hamu, N. Shamir, and Y. Lipman, Invariant and equivariant graph networks, ICLR (2019).
- [61] S. Luo, T. Chen, and A. S. Krishnapriyan, Enabling efficient equivariant operations in the fourier basis via gaunt tensor products, arXiv preprint arXiv:2401.10216 (2024).
- [62] A. Jain, S. P. Ong, G. Hautier, W. Chen, W. D. Richards, S. Dacek, S. Cholia, D. Gunter, D. Skinner, G. Ceder, *et al.*, Commentary: The materials project: A materials genome approach to accelerating materials innovation, *APL Mater.* **1**, 011002 (2013).
- [63] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, and J. Liang, Convolutional neural networks for medical image analysis: Full training or fine tuning?, *IEEE Trans. Med. Imaging* **35**, 1299 (2016).

- [64] M. I. Jordan and T. M. Mitchell, Machine learning: Trends, perspectives, and prospects, *Science* **349**, 255 (2015).
- [65] H. Zhang, G. Li, J. Li, Z. Zhang, Y. Zhu, and Z. Jin, Fine-tuning pre-trained language models effectively by optimizing subnetworks adaptively, *Adv. Neural Inf. Process. Syst.* **35** (2022).
- [66] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in *Proc. 2019 Conf. North Am. Chapter Assoc. Comput. Linguist.: Hum. Lang. Technol.* (2019) pp. 4171–4186.
- [67] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang, Biobert: a pre-trained biomedical language model for biomedical text mining, *Bioinformatics* **36**, 1234 (2020).
- [68] S. Kim, H. Yang, Y. Kim, Y. Hong, and E. Park, Hydra: Multi-head low-rank adaptation for parameter efficient fine-tuning, *Neural Netw.* **178**, 106414 (2024).
- [69] H. Kaur, F. Della Pia, I. Batatia, X. R. Advincula, B. X. Shi, J. Lan, G. Csányi, A. Michaelides, and V. Kapil, Data-efficient fine-tuning of foundational models for first-principles quality sublimation enthalpies, *Faraday Discussions* **256**, 120 (2025).
- [70] M. Radova, W. G. Stark, C. S. Allen, R. J. Maurer, and A. P. Bartók, Fine-tuning foundation models of materials interatomic potentials with frozen transfer learning, *npj Computational Materials* **11**, 237 (2025).
- [71] C. Wang, S. Hu, G. Tan, and W. Jia, Elora: Low-rank adaptation for equivariant gnns, in *Forty-second International Conference on Machine Learning* (2025).
- [72] J. Moellmann and S. Grimme, DFT-D3 study of some molecular crystals, *J. Phys. Chem. C* **118**, 7615 (2014).
- [73] P. Novelli, L. Bonati, P. J. Buigues, G. Meanti, L. Rosasco, M. Parrinello, and M. Pontil, Fine-tuning foundation models for molecular dynamics: A data-efficient approach with random features, *Proc. Adv. Neural Inf. Process. Syst.* (2024).
- [74] M. Kulichenko, B. Nebgen, N. Lubbers, J. S. Smith, K. Barros, A. E. Allen, A. Habib, E. Shinkle, N. Fedik, Y. W. Li, *et al.*, Data generation for machine learning interatomic potentials and beyond, *Chem. Rev.* **124**, 13681 (2024).
- [75] AISSquare, AISSquare Datasets (2025), accessed: April 2, 2025.
- [76] Z. Fan, NEP dataset repository (2025), accessed: April 2, 2025.
- [77] P. Ying, C. Qian, R. Zhao, Y. Wang, F. Ding, S. Chen, and Z. Fan, Adv. neural inf. process. syst.modeling complex materials: The rise of neuroevolution potentials, *arXiv preprint arXiv:2501.11191* (2025).
- [78] F. Eriksson, E. Fransson, and P. Erhart, The hiphive package for the extraction of high-order force constants by machine learning, *Adv. Theory Simul.* **2**, 1800184 (2019).
- [79] C. van der Oord, M. Sachs, D. P. Kovács, C. Ortner, and G. Csányi, Hyperactive learning for data-driven interatomic potentials, *npj Comput. Mater.* **9**, 168 (2023).
- [80] D. Marx and J. Hutter, *Ab initio molecular dynamics: basic theory and advanced methods* (Cambridge University Press, 2009).
- [81] G. Kresse and J. Hafner, Ab initio molecular dynamics for liquid metals, *Phys. Rev. B* **47**, 558 (1993).
- [82] R. C. Bernardi, M. C. Melo, and K. Schulten, Enhanced sampling techniques in molecular dynamics simulations of biological systems, *Biochim. Biophys. Acta Gen. Subj.* **1850**, 872 (2015).
- [83] Y. I. Yang, Q. Shao, J. Zhang, L. Yang, and Y. Q. Gao, Enhanced sampling in molecular dynamics, *J. Chem. Phys.* **151**, 070902 (2019).
- [84] A. Zhu, S. Batzner, A. Musaelian, and B. Kozinsky, Fast uncertainty estimates in deep learning interatomic potentials, *J. Chem. Phys.* **158**, 164111 (2023).
- [85] M. Kellner and M. Ceriotti, Uncertainty quantification by direct propagation of shallow ensembles, *Mach. Learn.: Sci. Technol.* **5**, 035006 (2024).
- [86] W. Edeling, M. Vassaux, Y. Yang, S. Wan, S. Guillais, and P. V. Coveney, Global ranking of the sensitivity of interaction potential contributions within classical molecular dynamics force fields, *npj Comput. Mater.* **10**, 87 (2024).
- [87] S. Venturi, R. Jaffe, and M. Panesi, Bayesian machine learning approach to the quantification of uncertainties on ab initio potential energy surfaces, *J. Phys. Chem. A* **124**, 5129 (2020).
- [88] M. Wen and E. B. Tadmor, Uncertainty quantification in molecular simulations with dropout neural network potentials, *npj Comput. Mater.* **6**, 124 (2020).
- [89] A. R. Tan, S. Urata, S. Goldman, J. C. Dietschreit, and R. Gómez-Bombarelli, Single-model uncertainty quantification in neural network potentials does not consistently outperform model ensembles, *npj Comput. Mater.* **9**, 225 (2023).
- [90] L. Zhang, D.-Y. Lin, H. Wang, R. Car, and W. E, Active learning of uniformly accurate interatomic potentials for materials simulation, *Phys. Rev. Mater.* **3**, 023804 (2019).
- [91] J. S. Smith, B. Nebgen, N. Lubbers, O. Isayev, and A. E. Roitberg, Less is more: Sampling chemical space with active learning, *J. Chem. Phys.* **148**, 241733 (2018).
- [92] M. Jiang, T. Xing, E. Zio, and X. Zhu, A bayesian data-driven framework for aleatoric and epistemic uncertainty quantification in remaining useful life predictions, *IEEE Sens. J.* **24**, 42255 (2024).
- [93] J. Dai, S. Adhikari, and M. Wen, Uncertainty quantification and propagation in atomistic machine learning, *Rev. Chem. Eng.* **41**, 333 (2024).
- [94] F. Grasselli, S. Chong, V. Kapil, S. Bonfanti, and K. Rossi, Uncertainty in the era of machine learning for atomistic modeling, *arXiv preprint arXiv:2503.09196* (2025).
- [95] T. Wang, Y. Wang, J. Zhou, *et al.*, From aleatoric to epistemic: Exploring uncertainty quantification techniques in artificial intelligence, *arXiv preprint arXiv:2501.03282* (2025).
- [96] R. R. Rahaman, Uncertainty quantification and deep ensembles, *Adv. Neural Inf. Process. Syst.* **34** (2021).
- [97] D. Althoff, L. N. Rodrigues, and H. C. Bazame, Uncertainty quantification for hydrological models based on neural networks: the dropout ensemble, *Stoch. Environ. Res. Risk Assess.* **35**, 1051 (2021).
- [98] A. Olivier, M. D. Shields, and L. Graham-Brady, Bayesian neural networks for uncertainty quantification in data-driven materials modeling, *Comput. Methods Appl. Mech. Eng.* **386**, 114079 (2021).
- [99] Y. Gal, P. Koumoutsakos, F. Lanusse, *et al.*, Bayesian uncertainty quantification for machine-learned models in physics, *Nat. Rev. Phys.* **4**, 573 (2022).

- [100] J. Zhang, Modern monte carlo methods for efficient uncertainty quantification and propagation: A survey, *Wiley Interdiscip. Rev. Comput. Stat.* **13**, e1539 (2021).
- [101] H.-O. R. Bae, R. V. Grandhi, and R. A. Canfield, Epistemic uncertainty quantification techniques including evidence theory for large-scale structures, *Comput. Struct.* **82**, 1101 (2004).
- [102] P. Manfredi and R. Trinchero, A probabilistic machine learning approach for the uncertainty quantification of electronic circuits based on gaussian process regression, *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **41**, 2638 (2021).
- [103] R. Hart, L. Yu, Y. Lou, *et al.*, Improvements on uncertainty quantification for node classification via distance based regularization, *Adv. Neural Inf. Process. Syst.* **36** (2023).
- [104] S. Munikoti, D. Agarwal, L. Das, *et al.*, A general framework for quantifying aleatoric and epistemic uncertainty in graph neural networks, *Neurocomputing* **521**, 1 (2023).
- [105] V. Kuleshov, N. Fenner, and S. Ermon, Accurate uncertainties for deep learning using calibrated regression, in *ICML* (PMLR, 2018).
- [106] P. Pernot, Calibration in machine learning uncertainty quantification: beyond consistency to target adaptivity, *APL Mach. Learn.* **1**, 046121 (2023).
- [107] Y. Liu, L. Wang, W. Zhu, and X. Yu, Farthest point sampling in property designated chemical feature space as an effective strategy for enhancing the machine learning model performance for small scale chemical dataset, *Journal of Materials Informatics* **5**, 39 (2025).
- [108] A. P. Bartók, J. Kermode, N. Bernstein, and G. Csányi, Machine learning a general-purpose interatomic potential for silicon, *Phys. Rev. X* **8**, 041048 (2018).
- [109] A. D. Naghdi, F. Pellegrini, E. Küçükbenli, *et al.*, Neural network interatomic potentials for open surface nano-mechanics applications, *Acta Mater.* **277**, 120200 (2024).
- [110] D. E. Smirnova, A. Y. Kuksin, S. V. Starikov, and V. V. Stegaliov, A ternary eam interatomic potential for u–mo alloys with xenon, *Modell. Simul. Mater. Sci. Eng.* **21**, 035011 (2013).
- [111] J. Byggmästar, K. Nordlund, and F. Djurabekova, Gaussian approximation potentials for body-centered-cubic transition metals, *Phys. Rev. Mater.* **4**, 093802 (2020).
- [112] W. M. Haynes, ed., *CRC Handbook of Chemistry and Physics*, 97th ed. (CRC Press, Boca Raton, FL, 2017).
- [113] S. Plimpton, Fast Parallel Algorithms for Short-Range Molecular Dynamics, *Journal of Computational Physics* **117**, 1 (1995).
- [114] 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, R. Shan, M. J. Stevens, J. Tranchida, C. Trott, and S. J. Plimpton, LAMMPS - a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales, *Computer Physics Communications* **271**, 108171 (2022).
- [115] J. Tersoff, Empirical interatomic potential for carbon, with applications to amorphous carbon, *Physical Review Letters* **61**, 2879 (1988).
- [116] J. Tersoff, Modeling solid-state chemistry: Interatomic potentials for multicomponent systems, *Physical Review B* **39**, 5566 (1989).
- [117] T. Werder, J. H. Walther, R. L. Jaffe, T. Halicioglu, and P. Koumoutsakos, On the water-carbon interaction for use in molecular dynamics simulations of graphite and carbon nanotubes, *The Journal of Physical Chemistry B* **107**, 1345 (2003).
- [118] G. Kresse and J. Furthmüller, Efficient iterative schemes for ab initio total-energy calculations using a plane-wave basis set, *Physical Review B* **54**, 11169 (1996).
- [119] P. E. Blöchl, O. Jepsen, and O. K. Andersen, Improved tetrahedron method for Brillouin-zone integrations, *Physical Review B* **49**, 16223 (1994).
- [120] B. Hammer, L. B. Hansen, and J. K. Nørskov, Improved adsorption energetics within density-functional theory using revised Perdew-Burke-Ernzerhof functionals, *Physical Review B* **59**, 7413 (1999).
- [121] S. Grimme, S. Ehrlich, and L. Goerigk, Effect of the damping function in dispersion corrected density functional theory, *Journal of Computational Chemistry* **32**, 1456 (2011).
- [122] M. R. Uhlig, D. Martin-Jimenez, and R. Garcia, Atomic-scale mapping of hydrophobic layers on graphene and few-layer MoS<sub>2</sub> and WSe<sub>2</sub> in water, *Nat. Commun.* **10**, 2606 (2019).
- [123] D. Kim, D. S. King, P. Zhong, and B. Cheng, Learning charges and long-range interactions from energies and forces, arXiv preprint arXiv:2412.15455 (2024).
- [124] Y. Xiao, Y. Wang, S.-H. Bo, J. C. Kim, L. J. Miara, and G. Ceder, Understanding interface stability in solid-state batteries, *Nature Reviews Materials* **5**, 105 (2020).
- [125] T. Huang, W. Dong, F. Wu, *et al.*, Uncertainty-driven knowledge distillation for language model compression, *IEEE/ACM Trans. Audio Speech Lang. Process.* **31**, 2850 (2023).
- [126] K. Song, R. Zhao, J. Liu, *et al.*, General-purpose machine-learned potential for 16 elemental metals and their alloys, *Nat. Commun.* **15**, 10208 (2024).
- [127] A. Siddiqui and N. D. Hine, Machine-learned interatomic potentials for transition metal dichalcogenide alloys, *npj Comput. Mater.* **10**, 169 (2024).
- [128] Y. Chiang, T. Kreiman, E. Weaver, *et al.*, MLIP Arena: Advancing fairness and transparency in machine learning interatomic potentials through an open and accessible benchmark platform, in *ICLR* (2025).
- [129] F. Ekström Kelvinius, D. Georgiev, A. Toshev, *et al.*, Accelerating molecular graph neural networks via knowledge distillation, in *Adv. Neural Inf. Process. Syst.*, Vol. 36 (Curran Associates, Inc., 2023).
- [130] I. Amin, S. Raja, and A. Krishnapriyan, Towards fast, specialized machine learning force fields: Distilling foundation models via energy Hessians, arXiv preprint arXiv:2501.09009 (2025).
- [131] K. Schütt, O. Unke, and M. Gastegger, Equivariant message passing for the prediction of tensorial properties and molecular spectra, in *ICML* (PMLR, 2021).
- [132] M. J. Willatt, F. Musil, and M. Ceriotti, Feature optimization for atomistic machine learning yields a data-driven construction of the periodic table of the elements, *Phys. Chem. Chem. Phys.* **20**, 29661 (2018).
- [133] E. Wigner, *Group theory: and its application to the quantum mechanics of atomic spectra*, Vol. 5 (Elsevier, 2012).