SAND2022-12875

SANDIA REPORT

SAND2022-xxxxx
Printed September, 2022

Sandia National Laboratories

Accelerating Multiscale Materials Modeling with Machine Learning

Normand A. Modine, John A. Stephens, Laura P. Swiler, Aidan Thompson, Dayton J. Vogel, Lenz Fiedler, Attila Cangi, Sivasankaran Rajamanickam

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185
Livermore, California 94550

Issued by Sandia National Laboratories, operated for the United States Department of Energy by National Technology & Engineering Solutions of Sandia, LLC.

Available to DOE and DOE contractors from

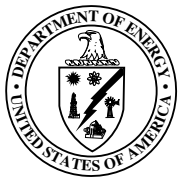
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@osti.gov
Online ordering: http://www.osti.gov/scitech

Available to the public from

U.S. Department of Commerce
National Technical Information Service
5301 Shawnee Road
Alexandria, VA 22312

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.gov
Online order: https://classic.ntis.gov/help/order-methods



Accelerating Multiscale Materials Modeling with Machine Learning

Normand A. Modine
Computational Materials and Data Science
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185
namodin@sandia.gov

John A. Stephens
Optimization and UQ
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185
jasteph@sandia.gov

Laura P. Swiler
Optimization and UQ
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185
lpswile@sandia.gov

Aidan Thompson
Computational Multiscale
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185
athomps@sandia.gov

Dayton J. Vogel
Computational Materials and Data Science
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185
dvogel@sandia.gov

Lenz Fiedler
CASUS
Matter under Extreme Conditions
02826 Görlitz
Germany
l.fiedler@hzdr.de

Attila Cangi
CASUS
Matter under Extreme Conditions
02826 Görlitz
Germany
a.cangi@hzdr.de

Sivasankaran Rajamanickam
Scalable Algorithms Department
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185
srajama@sandia.gov

ABSTRACT

The focus of this project is to accelerate and transform the workflow of multiscale materials modeling by developing an integrated toolchain seamlessly combining DFT, SNAP, LAMMPS, (shown in Figure 1-1) and a machine-learning (ML) model that will more efficiently extract information from a smaller set of first-principles calculations. Our ML model enables us to accelerate first-principles data generation by interpolating existing high fidelity data, and extend the simulation scale by extrapolating high fidelity data (10^2 atoms) to the mesoscale (10^4 atoms). It encodes the underlying physics of atomic interactions on the microscopic scale by adapting a variety of ML techniques such as deep neural networks (DNNs), and graph neural networks (GNNs).

We developed a new surrogate model for density functional theory using deep neural networks. The developed ML surrogate is demonstrated in a workflow to generate accurate band energies, total energies, and density of the 298K and 933K Aluminum systems. Furthermore, the models can be used to predict the quantities of interest for systems with more number of atoms than the training data set. We have demonstrated that the ML model can be used to compute the quantities of interest for systems with 100,000 Al atoms. When compared with 2000 Al system the new surrogate model is as accurate as DFT, but three orders of magnitude faster. We also explored optimal experimental design techniques to choose the training data and novel Graph Neural Networks to train on smaller data sets. These are promising methods that need to be explored in the future.

ACKNOWLEDGMENT

We are grateful for the support of this project through the Laboratory Directed Research and Development program at Sandia National Laboratories. The authors acknowledge the Computing and Information Sciences investment area, specifically the program manager, John Feddema. Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. We acknowledge several collaboration with students, postdoctoral researchers, and other collaborators. Particularly we would like to acknowledge James Fox, Austin Ellis, and Gabriel Popoola.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

SUMMARY

The focus of this project is to accelerate and transform the workflow of multiscale materials modeling by developing an integrated toolchain seamlessly combining DFT, SNAP, LAMMPS, (shown in Figure 1-1) and a machine-learning (ML) model that will more efficiently extract information from a smaller set of first-principles calculations. Our ML model enables us to accelerate first-principles data generation by interpolating existing high fidelity data, and extend the simulation scale by extrapolating high fidelity data (10^2 atoms) to the mesoscale (10^4 atoms). It encodes the underlying physics of atomic interactions on the microscopic scale by adapting a variety of ML techniques such as deep neural networks (DNNs), and graph neural networks (GNNs).

We developed a new surrogate model for density functional theory using deep neural networks. The developed ML surrogate is demonstrated in a workflow to generate accurate band energies, total energies, and density of the 298K and 933K Aluminum systems. Furthermore, the models can be used to predict the quantities of interest for systems with more number of atoms than the training data set. We have demonstrated that the ML model can be used to compute the quantities of interest for systems with 100,000 Al atoms. When compared with 2000 Al system the new surrogate model is as accurate as DFT, but three orders of magnitude faster. We also explored optimal experimental design techniques to choose the training data and novel Graph Neural Networks to train on smaller data sets. These are promising methods that need to be explored in the future.

NOMENCLATURE

Table 0-1.

Abbreviation	Definition
ADOS	Atom-Decomposed Density of States
DFT	Density Functional Theory
DOE	Department of Energy
GUI	Graphical User Interface
IAP	Interatomic Potential
LDOS	Local Density of States
MD	Molecular Dynamics
ML	Machine Learning
NN	Neural Network
OED	Optimal Experimental Design
PC	Principal Components
PCA	Principal Component Analysis
SNAP	Spectral Neighbor Analysis Potential

1. INTRODUCTION

Multiscale materials modeling (MMM) provides direct input to device-level simulations and experimental design. The atomistic scale is bridged to the device level on the macroscopic scale by (1) use of equation-of-state tables generated from atomistic (often first-principles) simulations and (2) through a sequence of input, output, and parameterization operations on a hierarchy of simulation tools.

Predictive atomistic materials modeling is hampered by costly and redundant generation of first-principles data. Large-scale molecular dynamics (MD) simulations (LAMMPS) [35] of material behavior require accurate interatomic potentials (IAPs) such as SNAP [46] that must be fit to higher fidelity datasets. These datasets are commonly generated using density functional theory (DFT), which is complex, expensive, limited to small scales (nanometers and femtoseconds), and exhibits O(N^3) scaling in system size. The amount of data needed to construct an IAP increases exponentially with the number of chemical elements, thermodynamic states, phases, and interfaces [49]. Likewise, evaluation of thermodynamic properties using DFT is computationally expensive, typically 10^5 to 10^6 core-hours for each point on a grid of temperatures and densities [48]. Furthermore, fully converged DFT calculations at low densities, high temperatures, or near phase transitions are even infeasible.

The focus of this project is to accelerate and transform the workflow of multiscale materials modeling by developing an integrated toolchain seamlessly combining DFT, SNAP, LAMMPS, (shown in Figure 1-1) and a machine-learning (ML) model that will more efficiently extract information from a smaller set of first-principles calculations. Our ML model enables us to accelerate first-principles data generation by interpolating existing high fidelity data, and extend the simulation scale by extrapolating high fidelity data (10^2 atoms) to the mesoscale (10^4 atoms). It encodes the underlying physics of atomic interactions on the microscopic scale by adapting a variety of ML techniques such as deep neural networks (DNNs), and graph neural networks (GNNs).

Pioneering efforts to apply ML to electronic structure calculations focus on mining benchmark databases generated from quantum chemistry methods and DFT in order to train ML models based on kernel ridge regression and feed-forward neural networks. These efforts enabled ML predictions of molecular properties [19, 37, 41] and crystal structures [42]. Only limited efforts have gone into actually speeding up DFT calculations by directly approximating solutions of the Kohn-Sham differential equations through ML models. Prior efforts have focussed on replacing the kinetic energy functional using ML regression [29, 44, 45] and deep-learning models [51]. Most recently, ongoing efforts have demonstrated the usefulness of ML kernel techniques on the electronic density [4] and feed-forward neural networks on the local density of states (LDOS) [7] for creating DFT surrogates. The ML methods bypass the expensive eigensolver step of DFT calculations for simple molecules (water, ethane, and malonaldehyde) and solids (aluminum and polyethylene). They have demonstrated that ML models of spatially resolved quantities (electronic density and LDOS) are able to reproduce the electronic structure at DFT-level accuracy.

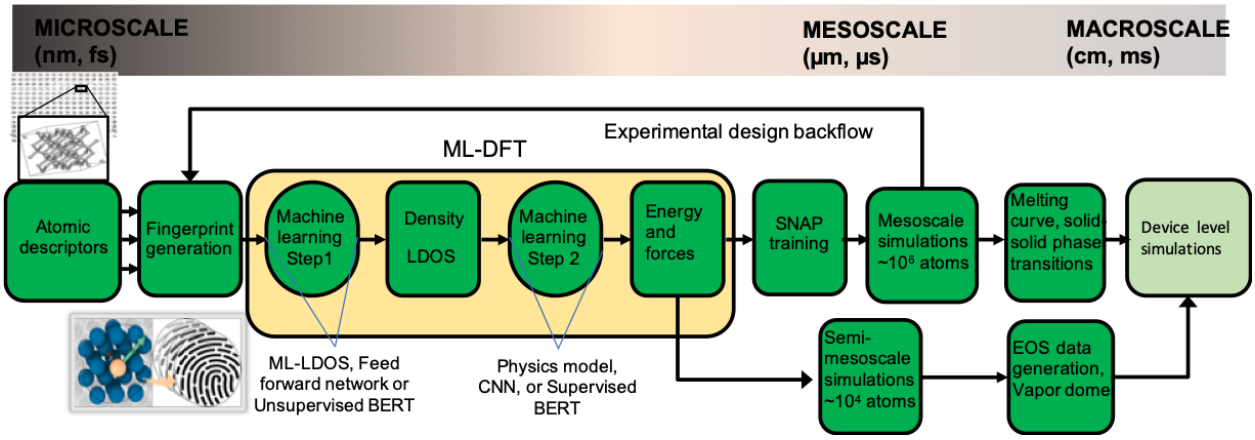


Figure 1-1. Overview of the Multiscale Materials Model Workflow

While our proposed MMM toolchain (Figure 1-1) is inspired by some of these breakthroughs, it distinguishes itself significantly from them: (1) instead of molecular systems, this project focused on materials relevant to Sandia's mission. For example, magnetohydrodynamics simulations at macroscopic length scales require accurate MD-based predictions of the energy and pressure. The LDOS based work that is close to our work [7] stopped short of demonstrating that ML can accurately calculate energies and forces needed to enable MD calculations; (2) this project represented materials descriptors (e.g., electronic density, LDOS) on a regular mesh as opposed to basis functions. This increased accuracy comes at increased memory cost. We alleviate this by scaling up on advanced architectures (3) this project also demonstrated the idea beyond data interpolation. Using the LDOS and the principle of near-sightedness in electronic structure theory [26], we will train and validate ML models using microscopic scale DFT training data (10^2 atoms), and then predict observables on the mesoscopic scale (10^4 atoms and higher).

The key contributions of this three year project are:

- A surrogate model for DFT calculation for Aluminum systems demonstrated for solid and liquid Aluminum [10].
- A scalable approach to generate fingerprints using the SNAP approach
- A high-fidelity training data generation approach with which we can train a good surrogate for DFT calculations
- A demonstration of robustness and scalability of the ML surrogate for DFT to systems up to 100,000 atoms
- A demonstration of up to three orders of magnitude speedup compared to traditional DFT calculations
- An experimental design approach to choose new training data
- An open source software framework, and a open ML model and training data allowed external groups to reproduce our results

The rest of the report is organized as follows. We described the foundational approach of this project a machine learning surrogate for density functional theory in Chapter 2. We describe briefly summarize an alternate approach using atom-centered density of states in Chapter 3. A new approach to experimental design using the atom-centered density of states is described in Chapter 4. We describe the software and conclude in Chapters 5 and 6 respectively.

2. MACHINE LEARNED DENSITY FUNCTIONAL THEORY (ML-DFT)

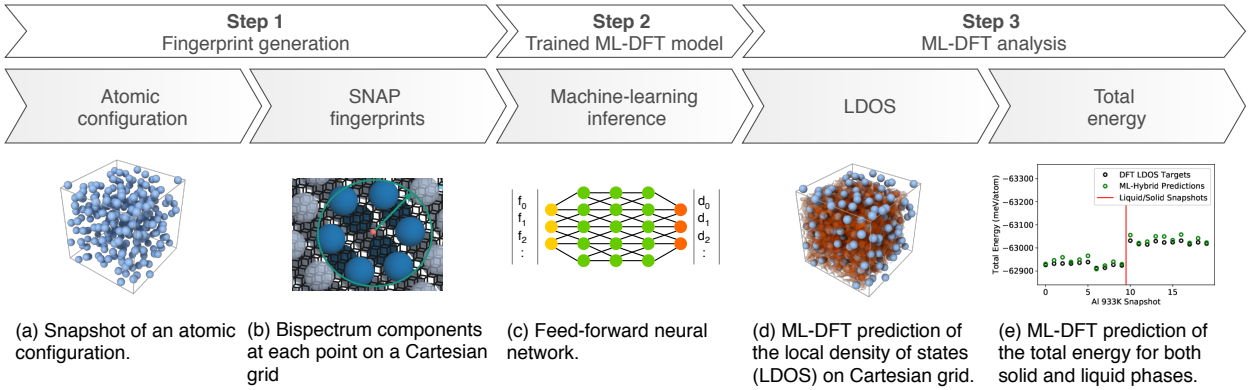
We focus on describing our method to develop a surrogate model for DFT calculations in this chapter. We extract relevant material from Ellis et al. [9] and add additional details as and when needed. The primary workflow in developing a surrogate model involves generating high fidelity training data (Section 2.1), generating fingerprints for grid points using a SNAP model (Section 2.2), train the ML model model to predict Local Density of States (LDOS) using the fingerprint as inputs (Section 2.3) and then computing band energies and total energies from LDOS (see [9]) or forces (Section 2.5).

2.1. Training Data Generation

The training data for our models was generated by calculating the LDOS for a set of atomic configurations using the Quantum Espresso electronic structure code [15, 16]. The structures were generated from snapshots of equilibrated DFT-MD trajectories for 256 atom supercells of aluminum. We developed training data for the solid phase of aluminum at ambient density (2.699 g/cc) at temperatures of 0K, 100K, 200K, 298K, 400K, 500K, 600K, 700K, and 933K and in expansion (2.4 g/cc) and compression (3.0 g/cc) at 298K and 933K. We also developed training data for liquid aluminum at 2.699 g/cc at 933K, 1000K, and 1100K. In most cases, training data was generated for three snapshots of the ionic positions, but for the 2.699 g/cc solid at 298K and 933K and all the liquid cases, we generated ten snapshots.

Our DFT calculations used a scalar-relativistic, optimized norm-conserving Vanderbilt pseudopotential (Al.sr-pbesol.upf) [18] and the PBEsol exchange-correlation functional [34]. We used a 100 Rydberg plane-wave cutoff to represent the Kohn-Sham orbitals and a 400 Rydberg cutoff to represent densities and potentials. These cutoffs resulted in a $200 \times 200 \times 200$ real-space grid for the densities and potentials, and we used this same grid to represent the LDOS. The electronic occupations were generated using Fermi-Dirac smearing with the electronic temperature corresponding to the ionic temperature. Monkhorst-Pack [30] k-point sampling grids, shifted to include the gamma point when necessary, were used to sample the band structure over the Brillouin zone. The DFT-MD calculations that generated the snapshots used $2 \times 2 \times 2$ k-point sampling for the crystalline phase and $1 \times 1 \times 1$ k-point sampling for the liquid phase. In contrast, the generation of the LDOS used $8 \times 8 \times 8$ k-point sampling. The reason for this unusually high k-point sampling (for a 256 atom supercell), as well as other details of the LDOS generation, is discussed in one of our published papers [9] In our calculations, we use an evenly spaced grid with a spacing of 0.1 eV, a minimum value of -10 eV, and a maximum value of +15 eV.

Using similar methods, we generated LDOS training data for the alpha-quartz, beta-quartz, beta-cristobalite, stishovite, tridymite, coesite, and seiferite crystal structures of SiO_2 at 0K, 100K, and 300K. We also calculated training data for the neutral and +1 oxygen vacancy in alpha-quartz at 0K and 300K. These calculations



(a) Snapshot of an atomic configuration.

(b) Bispectrum components at each point on a Cartesian grid

(c) Feed-forward neural network.

(d) ML-DFT prediction of the local density of states (LDOS) on Cartesian grid.

(e) ML-DFT prediction of the total energy for both solid and liquid phases.

Figure 2-1. Example of a Machine Learning-Density Functional Theory workflow taken from Ref.[9]

used the PBE exchange-correlation functional with the "O_ONCV_PBE-1.2.upf" and "Si_ONCV_PBE-1.2.upf" pseudopotentials. A 250 Rydberg plane-wave cutoff was used to represent the Kohn-Sham orbitals, while a 1000 Rydberg plane-wave cutoff was used to represent densities and potentials. The Brillouin zone was sampled with a $2 \times 2 \times 2$ k-point grid shifted to include the gamma point.

2.2. Fingerprint Generation

We assume that the LDOS at any point in space can be approximated by a function that depends only on the positions and chemical identities of atoms within some finite neighborhood of the point. In order to approximate this function using ML, we need to construct a fingerprint that maps the neighborhood of any point to a set of scalar values called *descriptors*. On physical grounds, good descriptors must satisfy certain minimum requirements: (i) invariance under permutation, translation, and rotation of the atoms in the neighborhood and (ii) continuous differentiable mapping from atomic positions to descriptors, especially at the boundary of the neighborhood. While these requirements exclude some otherwise appealing choices, such as the Coulomb matrix [38], the space of physically valid descriptors is still vast. In the context of ML-IAPs, the construction of good atomic neighborhood descriptors has been the subject of intense research. Recent work by Drautz [8] has shown that many prominent descriptors [2, 3, 46] belong to a larger family that are obtained from successively higher order terms in an expansion of the local atomic density in cluster integrals. The SNAP bispectrum descriptors [46] that we use in this work correspond to clusters of three neighbor atoms yielding four-body descriptors.

In contrast to prior work on ML-IAPs in which descriptors are evaluated on atom-centered neighborhoods, here we evaluate the SNAP descriptors on the same $200 \times 200 \times 200$ Cartesian grid points at which we evaluated the LDOS training data (see Eq. (**??**) above). For each grid point, we position it at the origin and define local atom positions in the neighborhood relative to it. The total density of neighbor atoms is represented as a sum of δ-functions in a three-dimensional space:

$$\rho(\mathbf{r}) = \delta(\mathbf{0}) + \sum_{r_k < R_{cut}^{\nu_k}} f_c(r_k; R_{cut}^{\nu_k}) w_{\nu_k} \delta(\mathbf{r}_k), \tag{2.1}$$

where $\mathbf{r}_k$ is the position of the neighbor atom k of element ν_k relative to the grid point. The w_ν coefficients are dimensionless weights that are chosen to distinguish atoms of different chemical elements ν, while

a weight of unity is assigned to the location of the LDOS point at the origin. The sum is over all atoms k within some cutoff distance $R_{cut}^{\nu_k}$ that depends on the chemical identity of the neighbor atom. The switching function $f_c(r; R_{cut}^{\nu_k})$ ensures that the contribution of each neighbor atom goes smoothly to zero at $R_{cut}^{\nu_k}$. Following Bartók *et al.* [2], the radial distance r_k is mapped to a third polar angle θ_0 defined by

$$\theta_0 = \theta_0^{max} \frac{r_k}{R_{cut}^{\nu_k}} . \tag{2.2}$$

The additional angle θ_0 allows the set of points $\mathbf{r}_k$ in the 3D ball of possible neighbor positions to be mapped on to the set of points (θ, ϕ, θ_0) on the unit 3-sphere. The neighbor density function can be expanded in the basis of 4D hyperspherical harmonic functions $\mathbf{U}^j$

$$\rho(\mathbf{r}) = \sum_{j=0,\frac{1}{2},\ldots}^{\infty} \mathbf{u}_j \cdot \mathbf{U}_j(\theta_0, \theta, \phi), \tag{2.3}$$

where $\mathbf{u}_j$ and $\mathbf{U}_j$ are rank $(2j+1)$ complex square matrices, $\mathbf{u}_j$ are Fourier expansion coefficients given by the inner product of the neighbor density with the basis functions $\mathbf{U}_j$ of degree j, and the $\cdot$ symbol indicates the scalar product of the two matrices. Because the neighbor density is a weighted sum of δ-functions, each expansion coefficient can be written as a sum over discrete values of the corresponding basis function

$$\mathbf{u}_j \quad = \quad \mathbf{U}_j(\mathbf{0}) + \sum_{r_k < R_{cut}^{\nu_k}} f_c(r_k; R_{cut}^{\nu_k}) w_{\nu_k} \mathbf{U}_j(\theta_0, \theta, \phi) . \tag{2.4}$$

The expansion coefficients $\mathbf{u}_j$ are complex-valued and they are not directly useful as descriptors because they are not invariant under rotation of the polar coordinate frame. However, scalar triple products of the expansion coefficients

$$B_{j_1 j_2 j} = \mathbf{u}_{j_1} \otimes_{j_1 j_2 j} \mathbf{u}_{j_2} \cdot (\mathbf{u}_j)^* \tag{2.5}$$

are real-valued and invariant under rotation [2]. The symbol $\otimes_{j_1 j_2 j}$ indicates a Clebsch-Gordan product of matrices of rank j_1 and j_2 that produces a matrix of rank j, as defined in our original formulation of SNAP [46]. These invariants are the components of the bispectrum. They characterize the strength of density correlations at three points on the 3-sphere. The lowest-order components describe the coarsest features of the density function, while higher-order components reflect finer detail. The bispectrum components defined here have been shown to be closely related to the 4-body basis functions of the Atomic Cluster Expansion introduced by Drautz [8].

For computational convenience, the LAMMPS implementation of the SNAP descriptors on the grid includes all unique bispectrum components $B_{j_1 j_2 j}$ with indices no greater than J_{max}. For the current study we chose $J_{max} = 5$, yielding a fingerprint vector consisting of 91 scalar values for each grid point. The cutoff distance and element weight for the aluminum atoms were set to 4.676 Å and 1.0, respectively.

2.3. Machine Learning Model

We use a feed-forward neural network with the inputs as the SNAP fingerprints at the real space grid point and the output vectors as the local density of states (LDOS) at the grid point. The LDOS values are generated as part of the training data generation described above using Quantum Espresso. Every

grid point with its SNAP fingerprint to LDOS mapping is treated as the independent training point. We use a grid of size 200x200x200 for 256 atom Aluminum systems. We use several snapshots at different temperatures from a DFT-MD simulation as the training data. To be specific, we use 10 snapshots of 298K Aluminum, 10 snapshots of liquid Aluminum at 933K and 10 snapshots of solid Aluminum at 933K as our training data. Each of this snapshots have 8M training points. We choose a small number of snapsots as training snapshots, one or two snapshots as validation data, and the balances for testing the ML model. We do not distinguish between points that belong to the same snapshot or different snapshots during training.

Table 2-1. Optimized hyper-parameters for the 298K and 933K models reproduced from [9].

Model Parameter	298K Model	933K Models
FP length	91	91
FP scaling	Elem.-wise stand.	Elem.-wise stand.
LDOS length	250	250
LDOS scaling	Max norm.	Max norm.
Optimizer	Adam	Adam
Minibatch size	1000	1000
Learning rate (LR)	1e-5	5e-5
LR schedule	4 epochs	4 epochs
Early stopping	8 epochs	8 epochs
Max layer width	800	4000
Layers	5	5
Activation Func.	LeakyReLU	LeakyReLU
Total weights	1.62e6	3.34e7

The ML-DFT models are built on top of the PyTorch framework and the Numpy and Horovod packages. All the software dependencies are described in detail in Chapter 5. The hyperparameters for the trained models for 298K and 933K models are shown in Table 2-1. These two models result in highly accurate predictions of the LDOS. We demonstrated the LDOS predicted using these models can be used to calculate total energies for both 298K and 933K Aluminum. Figure 2-2 shows one example of these results for band energy and total energy of the 298K Aluminum system. Overall, the hybrid models we developed by using both liquid and solid Aluminum for training and validation results in 13.04 meV/atom mean absolute total energy error for liquids and 31.60 meV/atom mean absolute total energy error for solids. The errors are even lower at 3.07 meV/atom mean absolute total energy error for 298K Aluminum model.

2.4. Scaling to larger systems

We have further extended the use of these ML models by training them on 256 atom system as before but using them for inference on systems that are up 100,000 atoms. Notice that ML inference is still on a single grid point in the real space going from SNAP fingerprint to LDOS at that grid point. As the number of atoms increase, we maintain a similar ratio of grid points and use the ML model in the inference mode to compute the LDOS. The evaluation of density, density of states, band energy and total energy all

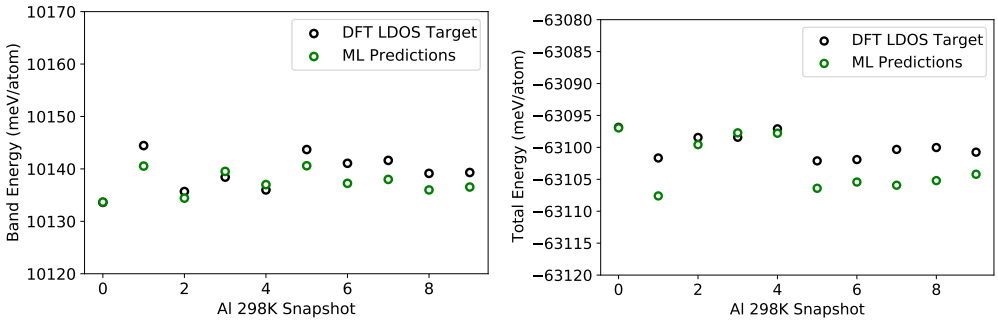


Figure 2-2. Band energy (left) and total energy (right) comparisons for the ML-DFT model trained on 1 solid snapshot at 298 K. The training and validation snapshot are snapshot 0 and 1, respectively (reproduced from [9]).

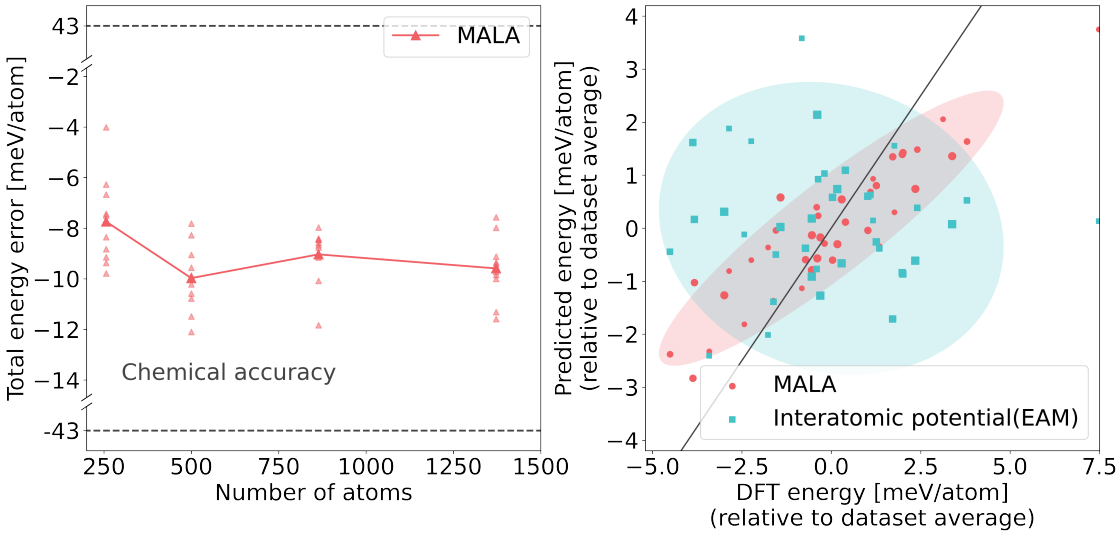


Figure 2-3. Total energy error when scaling the size of Aluminum systems starting from 256 atoms compared to DFT baseline. The chemical accuracy line is also shown (left). Training was done on 256 atom Al system. Comparison of DFT energy and predicted total energy for all the Aluminum systems with different number of atoms. The errors are more compact with our approach compared to EAM interatomic potential (right).

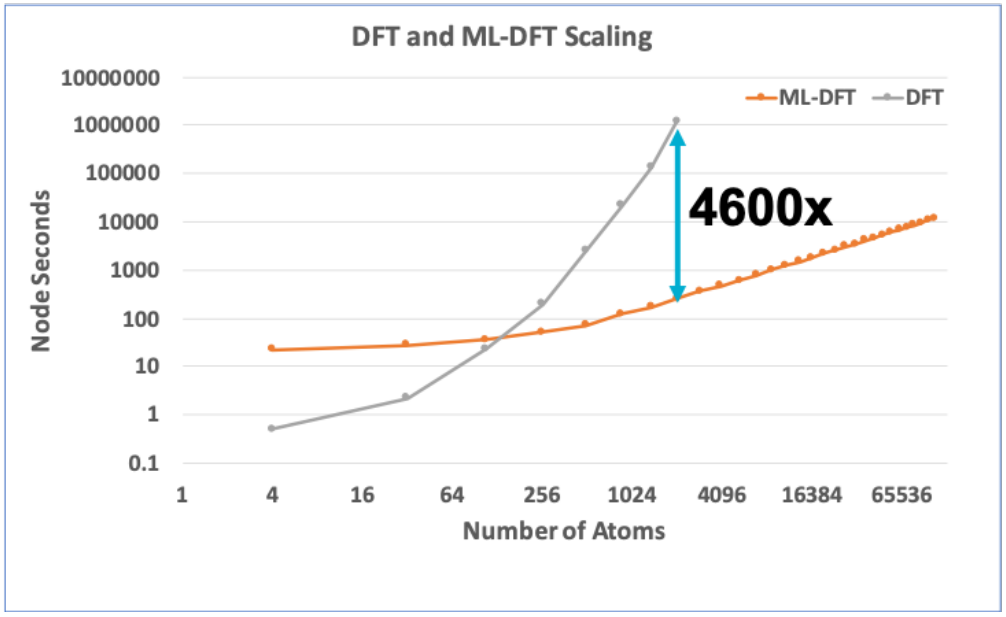


Figure 2-4. Comparison of time to compute band energy using standard DFT our method using machine learning for increasing system size from 4 atoms to 100,000 Aluminum atoms at 298K. The machine learning based approach scales linearly and results in three orders of magnitude speedup for a 2000 atom system.

follow naturally from there. Notice that this results in a method that is $O(n)$ in the number of atoms. We were able to demonstrate that the accuracy of the method compared to the traditional DFT up to 2,000 atoms (see Figure 2-3. As DFT scales $O(n^3)$ we were not able to run larger scale DFT calculations than this in the clusters we are using. When the compared with 2000 atom system ML-DFT is three orders of magnitude faster than DFT (see Figure 2-4). This was also made possible by several improvements to make the method both memory scalable and compute scalable.

2.5. Implementation of Forces

The forces are defined as the derivative of A^{BO} with respect to the atomic positions $\mathbf{R}$. A^{BO} depends on the density, the band energy, and the electronic entropy, which depend on $\mathbf{R}$ via the LDOS and the SNAP desciptors, and also has an explicit dependence on $\mathbf{R}$ via the Ewald energy and the atomic core density used in DFT non-linear core corrections.

The density, the band energy, and the electronic entropy are all functions of the LDOS, so the dependence of A^{BO} on these terms can be grouped together into a combined derivative with respect to the LDOS. This derivative is taken at constant electron number, so it is necessary to consider the change in the Fermi energy needed to keep the electron number constant when the LDOS changes and include the effects of this change on the density, band energy, and electronic entropy when calculating the derivative of A^{BO} with respect to the LDOS. Once this derivative has been calculated, the derivative of A^{BO} with respect to the SNAP descriptors can be calculated by back-propagation through the ML model, and then the forces can be calculated by contracting this derivative with the derivatives of the fingerprints with respect to the atomic positions. The later contraction can be done efficiently in LAMMPS by taking advantage of the

fact that the fingerprints at a point in space only depend on the atomic positions of the atoms that are within the cutoff distance of this point.

In order to avoid a scaling bottleneck in the computation of the structure factors in Quantum Espresso, the atomic positions are encoded into "Gaussian descriptors" in LAMMPS, and these Gaussian descriptors capture the explicit dependence of A^{BO} on **R**. Within this formalism, the force terms arising via the Gaussian descriptors can be treated exactly the same as the force terms due to the SNAP descriptors except that it is not necessary to back-propagate through the ML model.

The implementation of forces in MALA is approaching completion. Most of the required terms were coded in an earlier version of MALA, and "cached properties" were added to several objects in the newest version of MALA in order to avoid repeated calculation of the same quantities when calculating the forces. However, cached properties require a Python version more recent than 3.7, and it is only very recently that we have been able to build and run MALA at Sandia using these newer versions of Python. Once one remaining issue that is occuring when running the newest version of MALA at Sandia is resolved, we expect that it will take no more than a few weeks of effort to integrate the force terms that were previously coded into the newest version of MALA and to complete the implementation of the forces.

3. ATOM CENTERED DENSITY OF STATES

ML-DFT approach (Chapter 2) for accurately predicting the DOS through LDOS supervision involves millions of samples for resolving the DOS of a single configuration of atoms, which poses large computational demands. We also considered a novel atom-centered decomposition of DOS for supervision, which reduces the number of samples for training and evaluation by orders of magnitude compared to LDOS supervision. Combined with a new model for learning atomic environment descriptions end-to-end, our approach allows resolving downstream quantities such as band energy of melting point aluminum at a fraction of the cost of LDOS, with matching or greater accuracy. Our new ML model depends on a Graph Neural Network (GNN) approach to map atomic positions to an atom-centered density of states. We briefly describe these works here through excerpts from our past technical reports. More information can be found in the corresponding reports [12, 13].

3.1. Atom-Centered Density of States

In order to reduce the number of predictions that are required in order to evaluate the DOS for a given system, we wish to replace the LDOS $D^L(r, E)$ evaluated at grid points r and energies E with an "Atom-Decomposed Density of States" (ADOS) $D_i^A(E)$ evaluated for atoms i and energies E. There are two important requirements for the ADOS: (1) The DOS is given by a sum of the LDOS over grid points

$$D(E) = \sum_r D^L(r, E). \tag{3.1}$$

Summing the ADOS over atoms should produce the same DOS, i.e.,

$$\sum_i D_i^A(E) = D(E). \tag{3.2}$$

(2) If $D^L(r, E)$ can be accurately approximated as a function of the atomic positions in some local region around grid point r, then $D_i^A(E)$ can be accurately approximated by some function of the atomic positions in some local region around R_i, the position of atom i.

Both of the above properties can be achieved if $D_i^A(E)$ is defined as a weighted sum of $D^L(r, E)$ over grid points, and the weighted sum is a local partition of unity. In particular, if $D(r, E)$ is the LDOS evaluated at grid point r and energy E, we can define the ADOS associated with atom i as

$$D_i^A(E) = \sum_r w_i(r) D^L(r, E) \tag{3.3}$$

for some weighting functions $w_i(r)$. The set of weighting functions $w_i(r)$ is a partition of unity if

$$\sum_i w_i(r) = 1 \; \forall r. \tag{3.4}$$

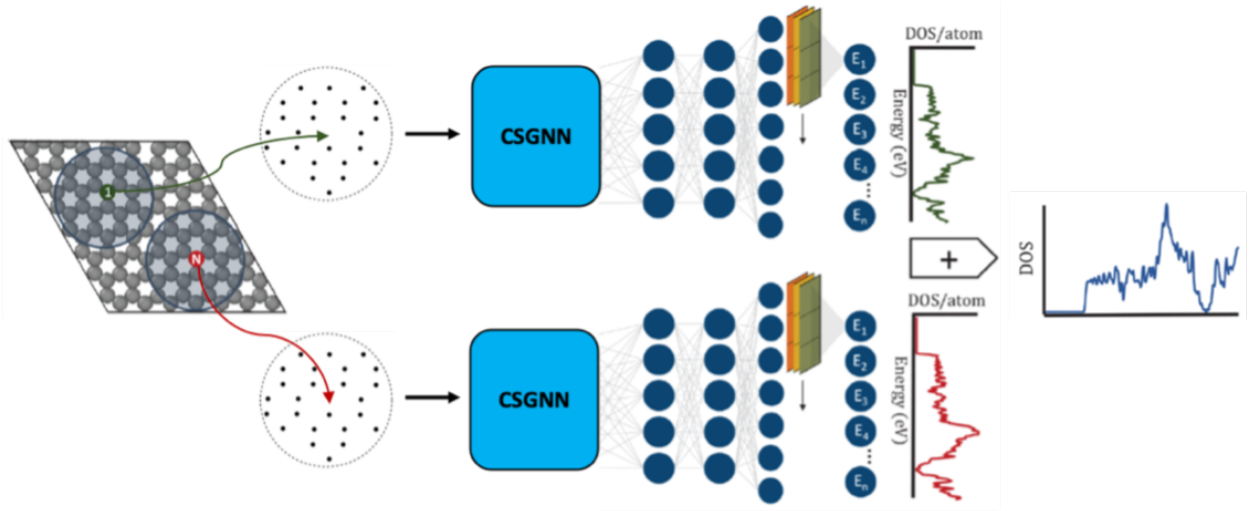


Figure 3-1. An illustration of the ADOS approach. A concentric sphere graph neural network (CSGNN) autoencodes atomic coordinates, and dense layers predict the atomic density of states (ADOS), which is then used to compute the band energy. Image taken from [12]

Likewise, the partition of unity is local if every $w_i(r)$ decays sufficiently rapidly for large $\|r - R_i\|$.

There are many way to define such a partition of unity, but the specific approach that we have chosen is to define

$$w_i(r) = \frac{\exp\left[-\|r - R_i\|^2/2\sigma^2\right]}{\sum_j \exp\left[-\|r - R_j\|^2/2\sigma^2\right]}. \tag{3.5}$$

Given this definition, it is easy to verify that $w_i(r)$ is a partition of unity and that Requirement (1) above is satisfied. For atom positions R_i that are evenly distributed throughout space, $w_i(r)$ decays as a Gaussian tail for large $\|r - R_i\|$, and the partition of unity is local. For systems that involve large regions with no atoms, some of the weighting functions $w_i(r)$ can remain substantial throughout such regions. However, $D^L(r, E)$ is generally small in such regions, at least for energies E that are occupied by electrons, and thus, for practical purposes, we believe that Requirement (2) also holds for such systems.

When σ is much less than the distance between atoms, the partition of unity defined above closely approximates an approach in which the LDOS at each grid point is assigned to the nearest atom. In the opposite limit, which σ is comparable to the distance between atoms, the LDOS at each grid point is shared between several atoms. We have picked an intermediate value of σ = 1.3 Angstroms, compared to an average nearest neighbor distance of around 2.6 Angstroms. Thus, grid points near to an atom will mostly have their LDOS assigned to that atom, while grid points between atoms will have their LDOS shared between the nearby atoms.

Using the above approach, we calculated the ADOS from our previously evaluated LDOS in order to generate training data for a model that predicts the ADOS as a function of the local environment around each atom. This model can then be used to predict the ADOS directly while avoiding the computationally expensive evaluation of the LDOS.

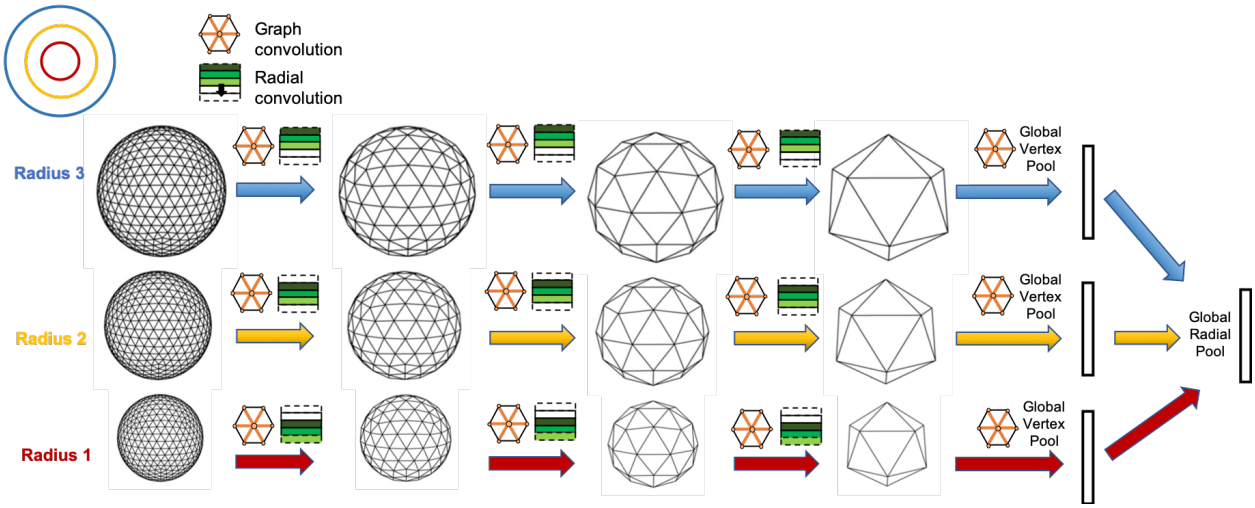


Figure 3-2. Example CSGNN architecture with $R = 3$ concentric spheres. Graph convolutions are followed by radial convolutions at each density of spherical sampling. Graph convolution is applied within each sphere. 1D convolution is applied between co-radial vertices (3 in this example). Vertex pooling (not shown) and downsampling then coarsens the spherical sampling. Global pooling is applied at the end to obtain the final feature representation. Image taken from [12]

3.2. Rotation equivariant Graph Neural Networks

Existing ML approaches for resolving DOS have so far relied on hand-crafted descriptors to extract features (*fingerprints*) from local atomic environments, as the input to their ML model. While much progress has been made in the development of fingerprinting techniques, they share in common the constraint of being limited to fitting to fixed basis functions. This work proposes to instead use trainable neural descriptors for fingerprinting, specifically focusing on the Concentric Spherical GNN (CSGNN) model [12, 13] as extended to the DOS prediction problem. This allows the atomic environment fingerprinting to be adapted to the data and target problem, with the end goal of generalizing to greater types and complexities of environments within a single model.

A workflow of the overall ADOS ML approach is illustrated by Fig. 3-1. CSGNN, the proposed model, operates on a concentric spherical spatial sampling of 3D space. Each individual sphere is discretized by the icosahedral grid, resulting in a highly uniform sampling of spherical space. The grid is sub-divided recursively to create higher sampling resolution. The sampling is further extended radially, resulting in concentric spheres about a center, which is defined naturally as an atom for the ADOS problem. We refer to Fig. 3-2 for illustration of the concentric spherical grids. An atom's atomic environment is contained within the concentric spherical sampling, and mapped to an initial description over the sampling. Fig. 3-3 provides an illustration of this mapping.

Two types of convolutions are defined for representation learning over the concentric spherical grid: intra-sphere and inter-sphere convolutions. The former is implemented by graph convolutions [25], with connectivity defined by each vertex's local neighborhood in the icosahedral discretization. Inter-sphere convolutions operate between co-radial vertices, orthogonally to intra-sphere convolutions. The combined use of the two convolution types permits extracting of features volumetrically over the concentric spherical sampling. Furthermore, the intra-sphere convolutions are by design rotationally equivariant to

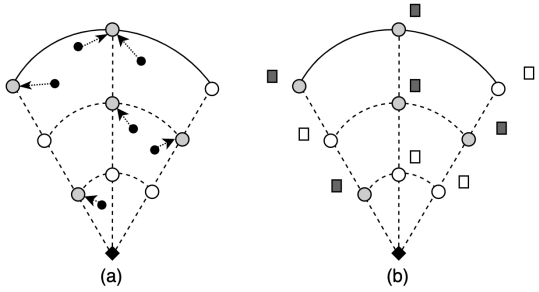


Figure 3-3. Shown is a 2D cross section of an atomic environment centered at a reference point (black diamond), for an example sector. (a) Each atom (black dot) in the environment a value of $\phi(r)$ to its nearest vertex in 3D space, where r is radial distance from the center and ϕ is a chosen distance mapping (such as the inverse function). (b) Values incident at any given vertex are summed, resulting in a scalar input feature per vertex. Image taken from [12]

the icosahedral rotation group [50], and approximately equivariant to the general space of 3D rotations. We refer to [13] for more detailed discussion of the concentric spherical convolutions. We combine the proposed convolutions into a hierarchical convolutional architecture, by also utilizing pooling and downsampling over the icosahedral grid. Fig. 3-2 illustrates an example CSGNN architecture. Convolutions at different scales of spherical sampling enables learning representation of the input atomic environment analogously to 2D CNNs for images.

3.3. Key Results

We summarize the key results of this work here and refer the reader to the longer technical report [12] for the full set of results.

The ADOS approach successfully reconstructed the original DOS derived from LDOS. We further verified that the band energy derived from ADOS matches the original band energy. The ADOS approach used 32,000 times fewer training points and was still able to match the accuracy needed of the LDOS based approach. ADOS based approach provides a 240× speedup in training per epoch and 54× speedup in inference compared to LDOS based approach.

4. OPTIMAL EXPERIMENTAL DESIGN

This chapter summarizes the experimental design activities undertaken as part of this LDRD. The focus of the experimental design activities is to support predictions made in a machine learning (ML) approach to material modeling as outlined in [9]. In particular, the goal is to develop approaches which identify the next set of atomic configurations which should be run through the DFT calculations to generate more training data for the workflow shown in Figure 2-1 *to improve the predictions of total energy*.

The workflow shown in Figure 2-1 is reproduced from [9]. In this workflow, snapshots of atomic configurations are the initial inputs. From these, bispectrum components are generated which become features for a neural network model which is used to predict local density of states (LDOS), which then feed into a prediction of total energy. The entire process is large scale: the density functional theory (DFT) calculations are highly resolved spatially (e.g. 8 million Cartesian grid points in an atomic configuration). At each gridpoint, there is a "fingerprint" involving 91 bispectrum components which are atomistic descriptors; these 91 components are mapped to 250 LDOS values at that point. The ML approach involves a vector-to-vector mapping. The multiple stages of the workflow and associated translation/aggregation of information at each stage along with noise and errors in the process makes this a challenging framework for experimental design.

Section 4.1 provides an overview of experimental design. It reviews conventional methods and explains why these approaches are inadequate for this workflow. Section 4.2 presents results for some alternative approaches that were investigated in this LDRD. Section 4.3 presents recent results which are customized for the neural network method used in ML: these results rely on dropout and deep ensembles, both of which attempt to characterize uncertainty in neural network model predictions.Then, this uncertainty can be calculated per candidate configuration and the configuration(s) with the largest prediction uncertainty are identified as the next candidates to add to the training set. Section 4.4 presents some concluding remarks and potential future directions.

4.1. Experimental Design

4.1.1. Definitions

Experimental Design refers to the selection of parameter settings at which to run physical or computational experiments. Statisticians classify design approaches into classical Design of Experiment (DoE) methods [31] and design and analysis of computer experiments (DACE) methods [40]. As an example of a parameter setting (sometimes called a factor setting), we will use a simple example with parameter A and B. Parameter A is binary with values $\{0, 1\}$ and parameter B has three levels $\{0.2, 0.9, 1.5\}$. One parameter setting might be $\{1, 0.2\}$ in which parameter A has a value of 1 and parameter B has a value of 0.2. If the parameter space is small enough, one can perform a "full factorial" design which covers all the parameter combinations.

An experimental design is often specified with a design matrix, X, where the number of rows of X are the number of runs (or number of samples in computational experiments) and the columns of X are the parameters or factors. A full factorial design for this simple example would involve 2 levels of parameter A times 3 levels of parameter B for 6 runs.

Classical DoE techniques arose from technical disciplines that assumed some randomness and nonrepeatability in field experiments (e.g., agricultural yield, experimental chemistry). In these cases, it is necessary to run replications, running the same set of experimental settings multiple times (called replicates) to see how the response varies within that setting. A common example is that of crop yields, where the parameter of interesting might be the application of fertilizer (on/off). A "replicate" would be one of several plots to which fertilizer was applied or one of several plots to which it was not applied. The replication is needed when there can be significant variation within a treatment (e.g. combination of experimental settings). DoE approaches such as central composite design, Box-Behnken design, and structured factorial designs have approaches to generate and handle replicate runs. These designs also put sample points at the extremes of the parameter space, since such designs offer more reliable trend extraction in the presence of nonrepeatability. NIST has an excellent handbook about experimental designs [1].

DACE methods are distinguished from DoE methods in that the nonrepeatability component is omitted for computer simulations which are deterministic (e.g. one set of input parameters always results in the same output. This is usually the case for the partial differential equations models used to solve physical problems). Thus, for DACE experiments, there are no replicates. In these cases, space-filling designs and Latin hypercube sampling are more commonly employed to accurately extract trend information. Quasi-Monte Carlo sampling techniques which are constructed to fill the unit hypercube with good uniformity of coverage can also be used for DACE. Space filling designs are also employed when constructing surrogate models, and much of the early DACE work centered around sampling to construct Gaussian process models. [39, 40, 43]

4.1.2. Optimal Experimental Design

The selection of input parameters for design of experiments may be done in many ways. First, one needs to consider the question being addressed [1, 40]: is the purpose of the design to choose option A or B (comparative designs) or identify important parameters (screening designs)? Or is the purpose of the design to build a response surface model such as a regression model, in which case one might want to minimize the prediction variance from the model or optimize coefficients in the model, etc.

There are several criteria one can choose to optimize when selecting a design. For example, Monte Carlo sampling over the parameter domain generally tries to select points with good "space filling" properties. There are several "alphabet-optimal" designs such as A-optimal, B-optimal, D-optimal, G-optimal, and I-optimal. These designs all involve optimizing some property of the design matrix, X. The information matrix is denoted by the inverse of the variance matrix, or $X^T X^{-1}$. An A-optimal design minimizes the trace of the inverse of the information matrix which results in minimizing the average variance of the estimates of regression coefficients built on the dataset X. A D-optimal design minimizes the determinant of the information matrix which results in maximizing the information content of the model coefficient values (this is also has the effect of good "space filling" properties). A G-optimal design minimizes the maximum variance of the predicted values from a regression fit built on the dataset X.[40]

4.1.3. Adaptive or Sequential Designs

The above designs are fixed designs that seek to optimize a particular property. The alphabet-optimal designs outlined above assume that a linear regression model is fit. This leads to convenient analytic formulations. There are also adaptive designs or sequential designs in which an initial experiment is run and then an optimization procedure identifies the "next best" experiment to run to optimize some objective. Sequential designs are sometimes called closed-loop designs because the results of experiment k guides the design of experiment $k+1$. Huan and Marzouk [21] present a sequential design framework where they pose the optimization of the sequential design as a dynamic program. This provides a nice framework but can be very expensive to solve. Up front, one must define all possible states, utilities, and experiments that can be performed. The optimization is formulated as finding a set of policies which maximize an expected utility function. The policies specify which design should be chosen at step k conditional on the state at that point, where $k = 1 \dots K$ and K is the maximum number of experiments to be performed.

4.1.4. Bayesian Experimental Design

As mentioned, the objective of optimal experimental design often involves choosing parameter sets which result in gaining as much information as possible. One example of this information gain is in Bayesian experimental design which has become popular over the past two decades. [6] Bayesian experimental design has the goal of determining experiments which "most inform" the posterior distribution inferred on model parameter values.

To explain Bayesian experimental design, we first need to explain Bayesian inference, which is a statistical process in which prior parameter information about parameters is informed by observational data to produce a posterior distribution. For example, we consider a model $\mathcal{M}$ with uncertain parameters θ. Bayes's theorem provides the posterior distribution of the model parameters θ given observational data y and prior distribution $p(\theta)$:

$$p(\theta|y) = \frac{p(y|\theta)p(\theta)}{\int p(y|\theta)p(\theta)d\theta}. \tag{4.1}$$

In addition to the prior, a key term in Equation 4.1 is $p(y|\theta)$ which is called the likelihood. This refers to the likelihood that the observed data y_i came from the model $\mathcal{M}(\theta)$. This likelihood is related to the discrepancy between the data and the model which is typically assumed to follow a Gaussian distribution.

$$y_i - \mathcal{M}(\theta_i) = {}_i \sim \mathcal{N}(0, \sigma^2) \quad i = 1, \dots, N_y. \tag{4.2}$$

The likelihood density is the product of the individual measurement errors:

$$\mathcal{L}(\theta) \equiv p(y|\theta) \equiv \frac{1}{(2\pi\sigma)^{N_y/2}} \exp\left(-\frac{1}{2\sigma^2} \|[y - \mathcal{M}(\theta)]\|^2\right). \tag{4.3}$$

To extend this to experimental design, the optimal parameters are now conditioned on the experimental design matrix, X. The Bayesian framework is extended as follows [6]:

$$p(\theta|y,X) = \frac{p(y|\theta,X)p(\theta)}{p(y|X)}. \tag{4.4}$$

where $p(y|X) = \int p(y|\theta,X)p(\theta)d\theta$.

The expected utility of an experiment X can then be calculated as:

$$E[U(X)] = \int p(y|X)U(y,X)dy \tag{4.5}$$

where $U(y,X)$ is a function of the posterior pdf $p(\theta|y,X)$.

One definition of the utility is the gain in Shannon information between the prior and posterior distribution:

$$U(y,X) = \int log(p(\theta|y,X))p(\theta|y,X)d\theta - log(p(\theta))p(\theta)d\theta \tag{4.6}$$

Another common utility function used is the Kullback-Leibler divergence of the prior from the posterior distribution, which is a measure of the differences in distributions [5]. Note that to maximize the expected utility in Equation 4.5, it is necessary to substitute Equation 4.6 for $U(y,X)$, resulting in a double integral. Note that the first term in Equation 4.5, $p(y|X)$, also requires an integral. The Bayesian formulations are very expensive to solve because of these nested integrals. Typically, one must employ sampling techniques such as Markov Chain Monte Carlo (MCMC) [17] or use variational methods as described in [11]. If one can exploit the structure of the problem, special methods can be employed as demonstrated in [24].

4.1.5. Challenges to Experimental Design

At least two challenges stand in the way of integrating the techniques of experimental design described in the previous sections into the workflow in Figure 2-1.

The first challenge is computational feasibility. Solving the nested integrals of Eq. 4.5 requires a large number of both model inference and model training calculations. In the present formulation of the problem, the computational cost of these is far too high for this approach to be practical.

The second challenge is physical in nature. The atomic positions in three dimensions fully determine the bispectrum components for a single-species "snapshot", and thus its predicted LDOS, band energy, and total energy. Physically relevant values of the atomic positions make up a vanishingly small fraction of the total "phase space" of permitted values. For example, a snapshot in which all of the atoms share exactly the same position is not a configuration of interest for training. Physical relevance places a severe constraint on generation of experimental designs using traditional DoE or DACE methods described above. Moreover, the input space of the neural network model is bispectrum components, not atomic positions. While we of course know how to compute the former from the latter, we do not know how to do the reverse.

Due to these challenges, we chose to pursue alternative approaches to selecting training data for the neural network.

4.2. Alternative Approaches

As explained, classical and Bayesian OED seemed not to be workable for the problems of generating or selecting additional training data or for estimating model uncertainty. To accomplish these tasks, we needed an alternative approach.

Intuitively, one ought to choose data for inclusion in the training set that have the following properties:

- It should be dissimilar to data that is already present.
- More than dissimilarity, we should have a reliable sense that LDOS predictions for the data will be poor. There's little reason to add data to the training set for which we can already make accurate predictions.
- Selection criteria should be based solely on the fingerprints, not on the DFT-predicted LDOS. We would like to avoid performing costly DFT calculations on candidate data, if possible.
- It should be fast.

With these criteria in mind, we sought to identify a measure of dissimilarity between fingerprints that correlated well with prediction error. Such a measure would provide not only a way to select data for inclusion in the training set but also a means of estimating the uncertainty of the model.

4.2.1. Principal Component Analysis

An immediate challenge of identifying a measure of dissimilarity between fingerprints is that the fingerprints are relatively high dimensional vectors. In our published work, we used 91-dimensional fingerprints. Our first step was to reduce the dimensionality of the fingerprint data.

Principal component analysis (PCA) is a widely used technique for dimension reduction. In PCA, an orthonormal basis for the data is constructed such that each basis vector (principal component) is selected to maximize the variation of the data projected onto it.[22] The data are first centered by subtracting the mean. Often, the first few principal components explain most of the variation in the data. Dimension reduction is achieved by truncating the set of basis vectors and representing the data in the new reduced basis.

The principal components are the eigenvectors of the data's covariance matrix, and principal component analysis can be accomplished using eigendecomposition techniques such as singular value decomposition.

In our case, the amount of data in the training set (20 snapshots with 8e6 fingerprints each) precluded straightforward SVD, and we employed an incremental decomposition approach implemented in the scikit-learn Python package [33]. On a single node of blake, it took a few minutes to update the PCA with information for each snapshot.

Figure 4-1 shows the fraction of variance in the fingerprint data for the 20 snapshots explained by the first six principal components. The first PC alone explains approximately 88.3% of the variance; the first two together 96.0%. The fingerprint data seemingly can be approximated using only a handful of principal components.

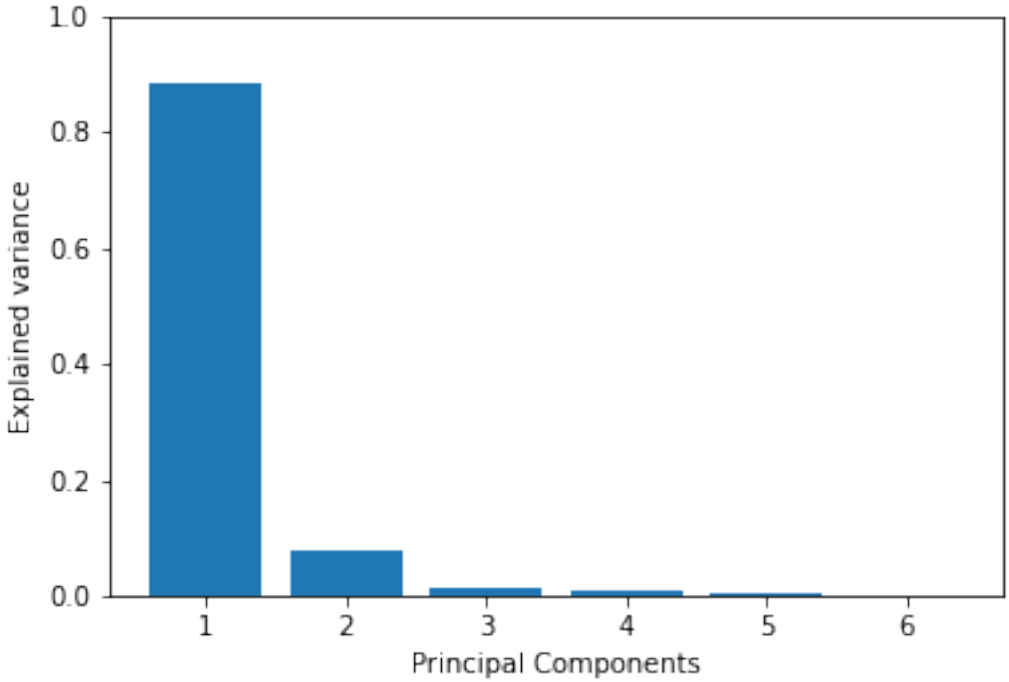


Figure 4-1. Fraction of fingerprint variance explained by the first six principal components

The PC scores are the projection of the data onto the orthonormal basis of the principal components. Figure 4-2 shows histograms of the scores for six example snapshots for the first and second principal components. The two topmost panes in the figure are for 298K solid snapshots 3 and 7. The middle two panes are for 933K liquid snapshots 3 and 7. And, the bottom two are for 933K solid snapshots 14 and 18.

For all six cases, the scores are characterized by a peak in which most of the scores are concentrated and a long, low tail. The 298K solids have the longest tails, followed by the 933K solids. The 933K liquids have visibly much shorter tails. One possible explanation for the phase-dependent tail length is that the solids, which are more crystalline than the liquids, have rarer high symmetry features. Between the 298K and 933K solids, the former have greater crystalline character, consistent with this explanation. The plausibility of this explanation increases our confidence that the principal components can provide insight into the underlying physics of the snapshots.

4.2.2. Measures of Dissimilarity in PC Space

With the principal component scores in hand, we began to explore measures of dissimilarity base upon them. The first few measures we examined compared entire candidate snapshots to snapshots already present in the training data. We also developed and considered a few that focused on individual fingerprints (fingerprints at specific spatial locations within snapshots).

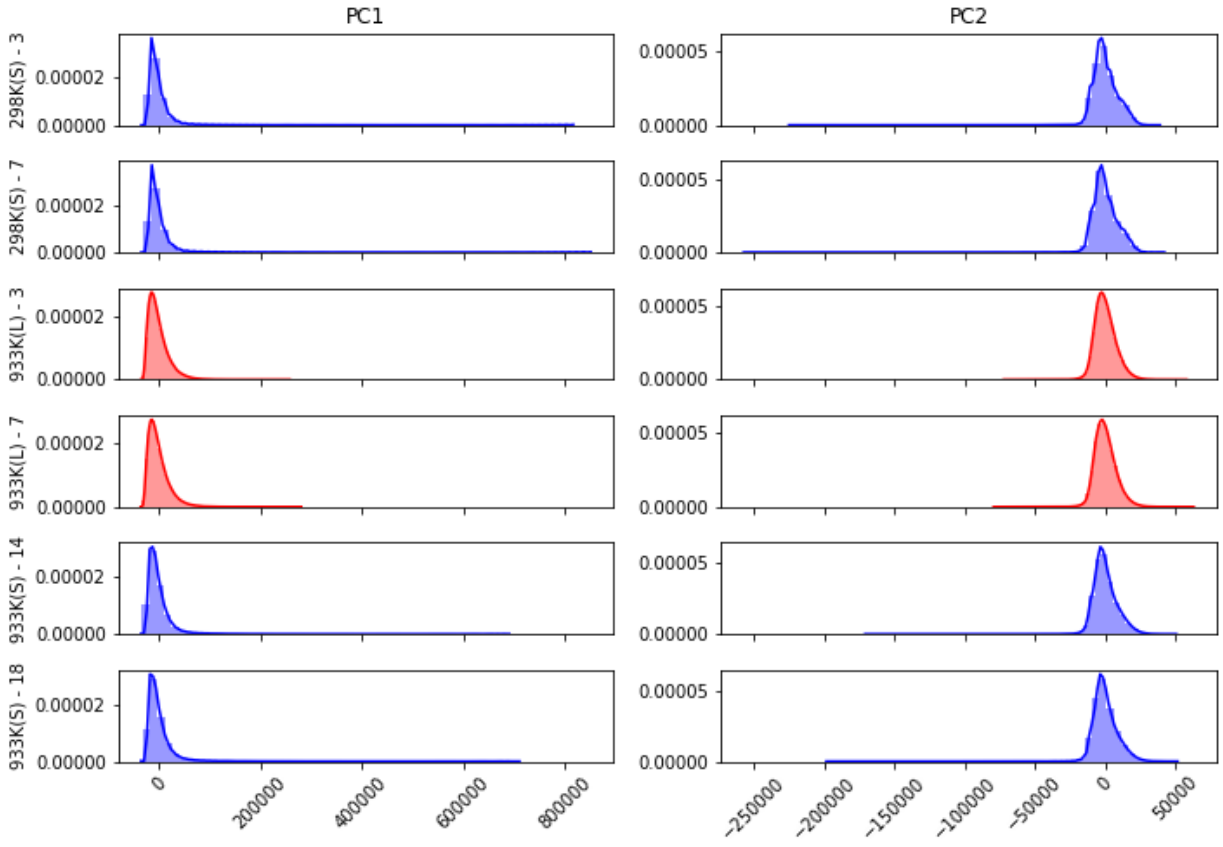


Figure 4-2. Projection of the fingerprint data for six example snapshots onto the first two principal components. The purple plots are for solid snapshots, and red are for liquid.

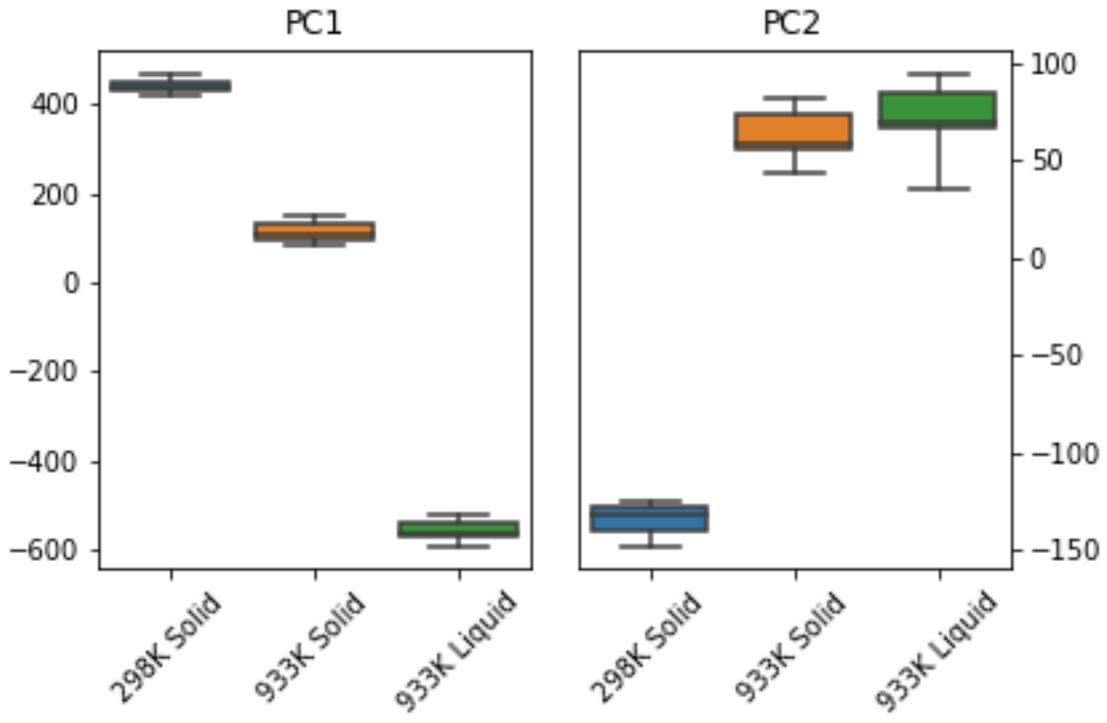


Figure 4-3. Mean principal component scores for the three phases in the training set.

4.2.2.1. Snapshot Based Measures

In this section, results for three different measures of dissimilarity between snapshots will be described. Then, correlations between these measures and LDOS prediction error will be presented.

For all three, results are based on comparisons between a training set consisting of 933K solid snapshots 10-13 and twenty candidate snapshots, the 298K solids (snapshots 0-9), 933K solids (snapshots 15-19) and 933K liquids (snapshots 0-9). LDOS prediction errors were the RMSE error between DFT predictions and those made by the 3rd model in Table II in [9].

These snapshots and model were selected because we believed they would make the least ambiguous demonstration of the overall approach. The 933K solid candidates should be most like the training data, followed by the 298K solids, followed by the 933K liquids. Moreover, a measure of dissimilarity that achieves this condition should also correlate well with prediction errors: the lowest error should be for the 933K solids, which the model has "seen", followed by worse error for the 298K solids, followed by 933K liquids, which are most dissimilar to the training data.

Differences in Means Figure 4-3 shows box plots of the mean first (PC1) and second (PC2) principal score for each of the twenty snapshots. To produce the plots, PC scores for each snapshot's 8e6 fingerprints were averaged. The box plots represent the distributions of these snapshot-level averages.

The evident differences between phases (298K solid vs. 933K liquid vs. 933K solid) in the distribution of PC scores shown in Figure 4-2 also appear in Figure 4-3, where the three phases can be easily distinguished.

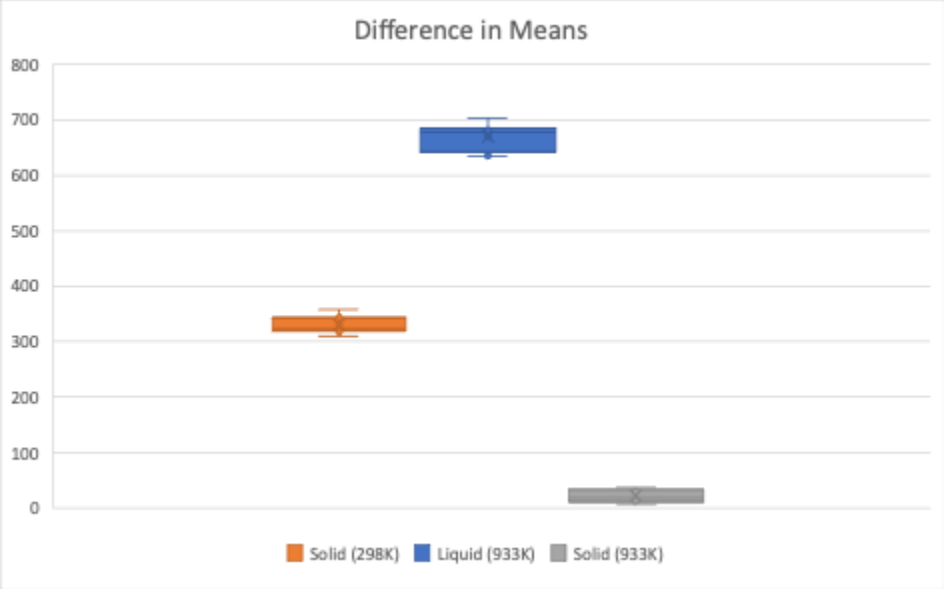


Figure 4-4. Difference in the mean PC1 score of candidate snapshots and training set snapshots.

This observation led us to consider a dissimilarity measure based on mean. Can the difference between the mean 1st principal component score of a candidate snapshot and the mean score for the training set snapshots be used to identify the next best snapshot to include in the training set?

Results initially were encouraging. Figure 4-4 shows these differences broken down by candidate snapshot phase. The training set contains 933K solid snapshots 10-14, and 933K solid candidate snapshots (15-19) are most similar (smallest difference in mean), followed by 298K solids (0-9). The 933K liquid snapshots (0-9) are the least similar. This is what we intuitively expect based on the physical similarities and differences between these three phases.

Wasserstein Metric The Wasserstein metric quantifies the difference between two probability distributions. If distributions are imagined as piles of sand, the Wasserstein metric may be thought of as the least amount of sand that must be moved to make the piles match.[36] Unlike other measures of differences between distributions such as the Kullback–Leibler divergence, the distributions are permitted to have different support.

Histograms of the 1st PC score of the fingerprints in each snapshot were first created. A separate histogram for the combined training set snapshots also was constructed. The Wasserstein metric was then computed between each of the candidate snapshot histograms and the training set histogram. We used the scipy implementation of Wasserstein distance for this purpose[47].

Figure 4-5 shows these distances by phase.

Like the difference-in-means metric, the ordering of the phases for the Wasserstein metric is what we might intuitively expect, both from the physics and by examination of the full distributions in Figure 4-2. The 933K solid candidates are most like (have the smallest distances from) the training set snapshots, which are also 933K solids. The second most similar are the 298K solids, and the least are the 933K liquids.

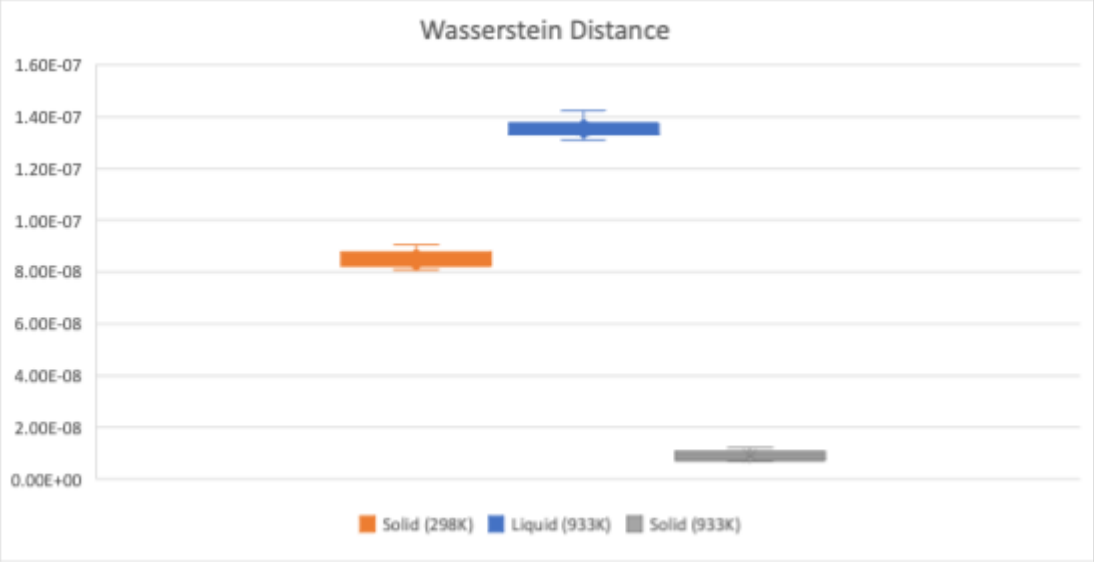


Figure 4-5. Wasserstein distance between candidate snapshots and training set snapshots.

PCA Misprediction The final measure of dissimilarity we considered is PCA misprediction. We define PCA misprediction as the RMSE error between actual fingerprints and fingerprints reconstructed from a truncated principal component basis. A fingerprint is reconstructed simply by multiplying the truncated basis matrix and the vector of its PC scores and adding the original mean.

Elsewhere, the principal components and scores we have considered were constructed from all twenty of the snapshots. In this section, we perform PCA on the fingerprints of only the training set snapshots. The motivation for this decision is the hypothesis that a truncated principal component basis created from the training set will result in lower misprediction for snapshots that are similar to the training set than for those that are not very similar.

Results are computing PCA misprediction for the candidate snapshots are shown in Figure 4-6. Compared to the difference-in-means and Wasserstein approaches, PCA misprediction is less able to distinguish 298K and 933K solids. However, it does capture the greater dissimilarity between the training set and 933K liquid candidates compared to the other two phases.

Distance-Error Correlation Having calculated three measures of snapshot-level dissimilarity, we now examine the relationship between them and LDOS prediction error. As stated previously, the prediction error is the RMSE between DFT-computed LDOS and the LDOS predicted by the 3rd model from Table II in [9], which was trained use 933K solid snapshots 10-13.

Although all three measures of dissimilarity were at least somewhat in agreement with intuition about differences between the training snapshots and candidate snapshots, none correlate well with LDOS prediction error. As shown in Figure 4-7, increases in the Wasserstein metric and difference-in-means correspond to increases in prediction error for the 933K and 298K solid phases, but the same is not true for the 933K liquid phase, which exhibit prediction errors lower than for the 298K solids. The PCA misprediction method also exhibits this problem and does not explain the difference in prediction errors for the solids. Even within individual phases, the difference-in-means is not positively correlated with

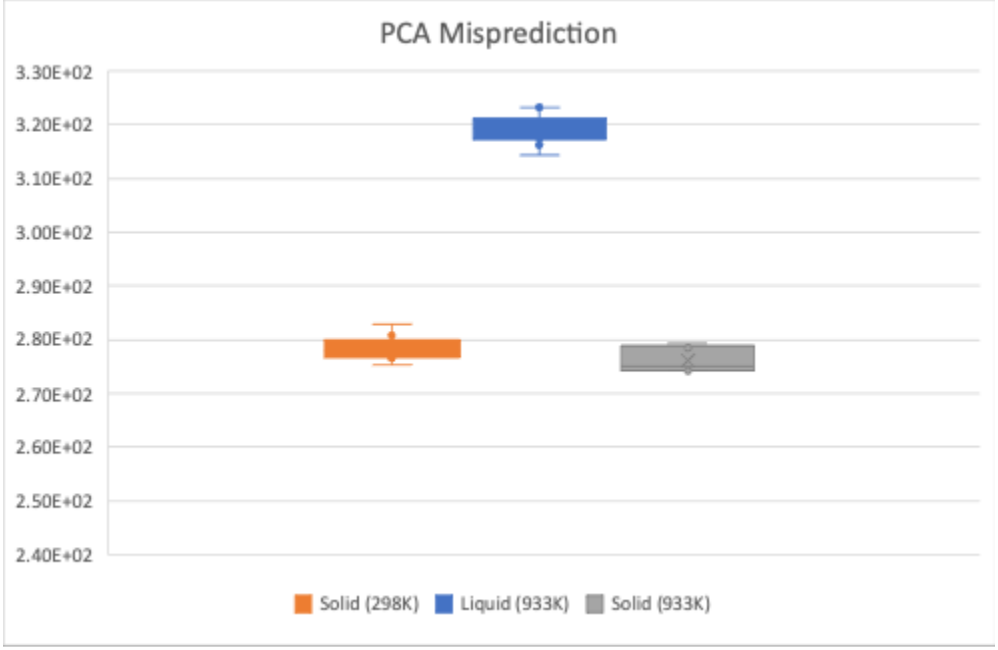


Figure 4-6. PCA misprediction by phase for candidate snapshots.

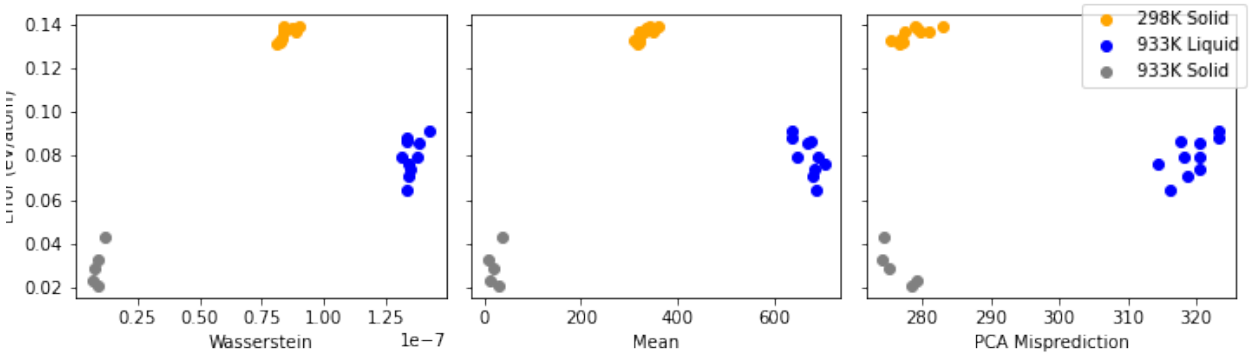


Figure 4-7. Relationship between LDOS prediction error and dissimilarity measures.

prediction error. While it is possible that the Wasserstein metric, which does not suffer from this last difficulty, may be of some use, more work would be needed to understand its limits.

4.2.2.2. Individual Fingerprint Based

In addition to measures of snapshot dissimilarity, we also considered two ways of measuring the dissimilarity between an individual fingerprint (i.e. a fingerprint at a single spatial location within a snapshot) and a set of other fingerprints. If successful, this style of approach would enable model training on partial snapshots, which potentially could reduce training time and redundancy in the training set.

Training Fingerprint Density The first method we attempted was based on fingerprint density. The PC score histograms in Figure 2 show that fingerprints are not uniformly distributed; some ranges of

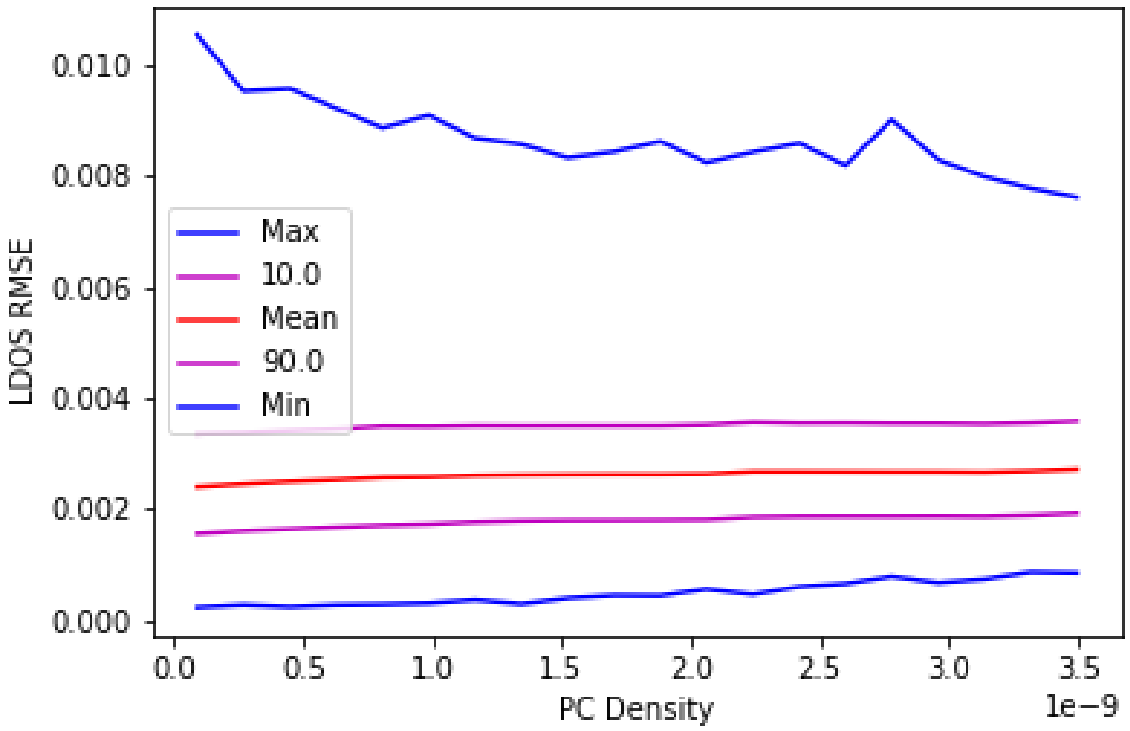


Figure 4-8. LDOS prediction error of candidate fingerprints versus their empirical principal component score density

values are more densely populated than others. In this method, candidate fingerprints that fall in ranges where training fingerprints are higher in frequency are considered more similar than those that fall outside such ranges. In somewhat plainer terms, if the histogram of training set fingerprint PC scores can be thought of as having an approximate probability distribution function $P(x)$, and x_c is PC score vector of a candidate fingerprint, then $P(x_c)$ indicates the degree of similarity between training set fingerprints and the finger that has x_c. Further, it should be correlated with the prediction error for that candidate fingerprint.

To test this hypothesis, we formed a normalized, two-dimensional histogram of the 1st and 2nd PC scores of the training set fingerprints. We then "looked up" the empirical density of candidate fingerprints in the histogram to determine their similarity to training set fingerprints. Lastly, we plotted LDOS prediction error versus these empirical density values by binning the densities and calculating statistics (minimum, 10th percentile, mean, 90th percentile, and maximum) over the bins.

The results for the fingerprints in one snapshot – 933K Liquid 6 – are shown in Figure 4-8. The expected relationship is that as density increases (candidate fingerprints become more similar to those that frequently occur in the training set), LDOS prediction error should decrease. Although the maximum statistic may exhibit this behavior, none of the others do. Surprisingly, the prediction error seems to be affected almost not at all by the empirical density. Although for brevity we present results only for the fingerprints of a single snapshot, others were examined the outcomes were similar.

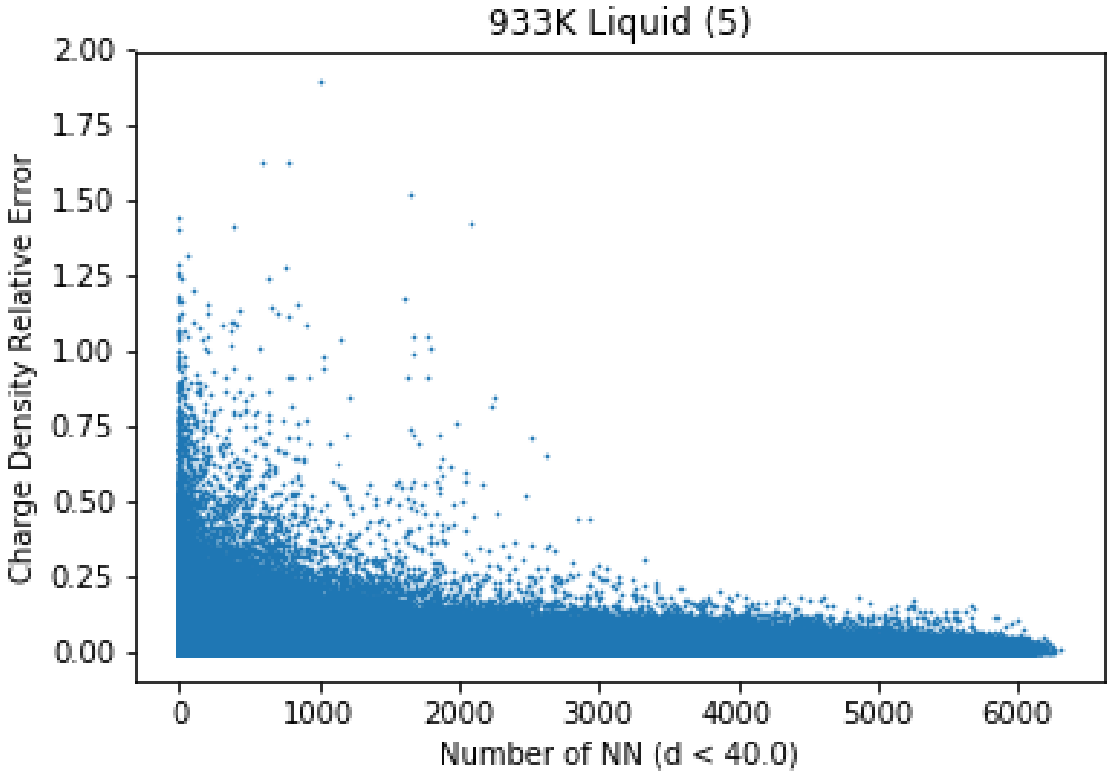


Figure 4-9. Relationship between the charge density prediction error of candidate fingerprints and the number of their nearest numbers in the training data.

Nearest Neighbors The second fingerprint-level approach we attempted was based on nearest neighbors. We searched for a relationship between prediction error and the number of neighbors within a small radius that (the PC scores of) a candidate fingerprint had among the fingerprints of the training data. We reasoned that the greater the number of a candidate fingerprint's nearest neighbors, the more similar it is to the training data. And, for this reason, its prediction error should decrease. Unlike the previous cases, which used the LDOS RMSE, we calculated the prediction error of the charge density. We used the first six principal components and weighted them by the variance that they explained.

The nearest neighbor calculations were performed using scikit-learn, using the Ball Tree algorithm [33].

As can be seen in Figure 4-9, our hypothesis appears to have been correct; as the number of nearest neighbors increases, the prediction error is reduced. This attempt to identify a relationship between prediction error and a measure of (dis)similarity was more successful than the others. However, it must be noted that the nearest neighbors calculation is costly (the fingerprints belonging to a single snapshot takes tens of minutes), and for this reason, the approach may not be very practical. We also emphasize that the nearest neighbors in Figure 4-9 are neighbors in the PCA space, not nearest neighbors in the space of bispectrum components directly.

4.3. Neural Network Uncertainty-Based Approach

Several of the above metrics considered for identifying "next best" candidates to add to the existing set of training data were not as fruitful as expected. Prediction uncertainty was the final metric considered in this project. The task of identifying good candidates for inclusion in the training set may be connected to the task of quantifying the uncertainty of deep neural networks. Techniques such as Monte Carlo drop-out, in which portions of the network are randomly deleted, have been explored to predict the uncertainty of the model for particular inputs. Such approaches may provide not only a warning of when a model's predictions may be insufficiently accurate but can be used to guide selection of candidate training data. Data for which model predictions are more uncertain may be a good candidate for additional training.

The idea is to augment the existing training data with candidate points that have large prediction uncertainty, typically modeled by the variance of the model prediction at the candidate point. This approach has roots in classical experimental designs. G-optimality involves minimizing the maximum variance of the predicted response while I-optimality involves minimizing the average variance of the predicted response over the entire design space.[40] The idea of using prediction variance has also been used extensively in adaptive Gaussian process methods [23], with the idea of adding the candidate point whose prediction variance is largest. In this section, we explore use of Monte Carlo dropout, a widely used method for assessing the uncertainty of ML models, for experimental design.

4.3.1. Monte Carlo Dropout

Monte Carlo dropout (or just dropout, for brevity) is a well-established technique in deep learning to reduce overfitting [20]. More recently, it has been suggested as a way to estimate the uncertainty of model predictions [14]. It is in this second capacity that we propose to use it for experimental design. The basic intuition is that test data for which model predictions are highly uncertain are good candidates for augmenting an existing training set, which can then be used to update the model. Importantly, test data need not be labeled to use Monte Carlo dropout. In the context of our present problem, this implies that we can use dropout to select snapshots before incurring the computational expense of calculating their LDOS using DFT.

Dropout is so named because it involves dropping out units of a model, such as nodes in the layers of a dense neural network (DNN). Consider first its use for regularization of a model. In this setting, dropout is employed during model training. At the beginning of each training epoch, nodes in the model are randomly deleted ("dropped") with a specified probability, called the dropout probability. Typical dropout probabilities are 10-50%. Weights and biases associated with dropped nodes are effectively frozen during that epoch and are not updated. They also do not contribute to model output, and outputs of the remaining nodes are weighted by one minus the dropout probability to maintain consistency. At the end of the epoch, the dropped nodes are returned to the network. At the beginning of the next epoch, all nodes are considered for dropout, independent of whether they have been dropped previously. The dropout procedure is illustrated in Figure 4-10.

For uncertainty quantification, dropout is used during inference. The dropout procedure itself is much the same as before. For a single test input, inference is performed many times. For each inference, a new submodel is derived from the full model by randomly dropping nodes and scaling the outputs of those

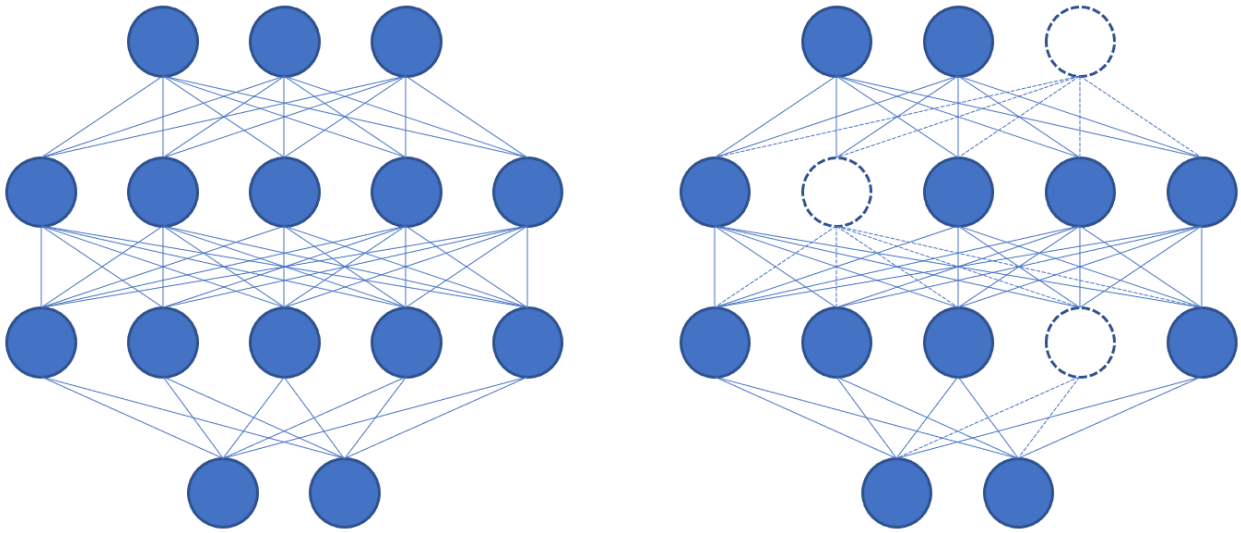


Figure 4-10. Illustration of dropout in a deep neural network. Inputs are at the top of the images and outputs at the bottom. The left image shows the fully connected network with no dropout. In the right image, three nodes have been dropped. Their associated weights, which have no effect, are shown as dashed lines.

that remain. Predictions from this ensemble of submodels are collected, and the standard deviation or variance of the predictions indicate the uncertainty of the model for the test input.

An advantage that dropout enjoys over another popular approach for uncertainty quantification in deep neural networks, deep ensembles [27], is its lower computational cost. In dropout, a single model is trained, and inference is run many times. The deep ensembles approach requires an ensemble of slightly different models to be trained, and inference to be run for each. It is for this reason that we chose to focus our efforts on dropout for this study.

4.3.2. Atom-Decomposed Density of States

Although the grid-based models used elsewhere in this work were successful at producing high-fidelity predictions of the LDOS, the cost of their training and inference would have impeded exploration of experimental design strategies. We chose to use an alternative, less expensive approach, which was developed by Fox, et al., termed the ADOS (atom-decomposed density of states) method [12].

In the ADOS method, instead of a spatial grid of atomic descriptors, the model inputs are the 3D atomic coordinates themselves. The label that the model predicts is the atom-decomposed density of states, which is the weighted sum over grid points of the LDOS, centered at each atomic nucleus. The reader is referred to [12] (especially Section 3.2) and [13] for full descriptions of the ADOS model and the underlying CSGNN (concentric sphere graph neural network); a basic illustration is reproduced in Figure 3-1. For the present purpose, the most important characteristic of the ADOS approach is that because the inputs are atomic coordinates rather than a dense spatial grid of large-dimensional atomic descriptors, it is far more convenient and less expensive to train and perform inference. Its most important limitation is that while the ADOS may be used to compute the band energy of a snapshot, it is not yet known how to compute the total energy, which is the quantity of interest for atomistic simulation. Our hope is that

despite this limitation, what we learn about experimental design using ADOS can be transferred to the LDOS model.

4.3.3. Approach and Results

This section describes our exploration of a potential experimental design technique that makes use of dropout and the ADOS model. We begin by describing the composition of the initial training sets we used. We then discuss hyperparameter optimization. Finally, we present inference results for test data and conclude by showing dropout uncertainty predictions.

4.3.3.1. Training Sets

The pool of data from which we drew our training sets included approximately 50 "snapshots" of DFT molecular dynamics (MD) trajectories of aluminum at 2.669 g cm-3. Each snapshot contained 256 atoms. The MD simulations were performed at a range of temperatures, from 0K to 1000K. Those below the melting temperature of Al (approximately 933K) are solid-like, and those above it are liquids. In addition, two MD simulations were performed at 933K, one in the solid phase and the other in the liquid. Ten snapshots were drawn from each of these. For our purposes, the data provided by these snapshots can be thought of as atomic positions and associated ADOS.

We considered three initial training sets. In the first and most important, we included four low-temperature (100K, 200K) solid snapshots as well as four high-temperature (933K) solid snapshots. No liquid snapshots were included. One low temperature and one high temperature snapshot were used for validation during training. We refer to this training set as *split-temperature*.

We selected temperatures at the two extremes of the range because we wanted to understand how errors and uncertainty would grow for solid phase test data as we moved inward in temperature. That is, we wanted to discover how well a model trained on such a training set could "interpolate" at intermediate temperatures.

Additionally, by including only solids in the split-temperature training set, we set up an easy initial test case for the idea of using dropout for experimental design. A model trained only on solids ideally should exhibit high dropout uncertainty for liquid test data. This expectation is based partly on previous experience with grid-based models, which struggled to make cross-phase predictions, and on our physical understanding: liquid snapshots substantially differ from solids in terms of atomic positions and energies and hence contain many "out of distribution" inputs. Therefore, if uncertainty for liquid snapshots is not clearly higher than for solids, it would tend to argue against using dropout uncertainty for experimental design.

The second training set we examined included only low temperature solid data (four 100K and 200K snapshots), and the third, only high temperature solids (four 933K snapshots). Results from models trained on these latter two training sets aided our interpretation of the first.

4.3.3.2. Hyperparameter Optimization

A full-factorial study of three hyperparameters was performed to identify the model parametrization that minimized the validation error for each of the three training sets. These hyperparameters were:

- **Learning rate**. This is the initial learning rate provided to the Adam optimizer, which was used for training. Permitted values were 0.01, 0.001, and 0.0001.
- **Number of hidden output layers**. In the ADOS model, the CSGNN autoencoder provides features to a set of dense layers. Given a parameterization of the CSGNN, the width of the layers is constant and determined by the output size of the CSGNN. The numbers of layers considered were 3, 4, 6, 8, and 10.
- **The parameter factor**. The parameter factor scales the number of output channels in the convolutional layers (Eq. 2 in [13]). Permitted values were 5, 6, 8, 10, and 12.

Three replicates were performed for each hyperparameter combination. The training was permitted to run for a maximum of 600 epochs, which was sufficient for the learning rate convergence criterion (1e-5) to be met in nearly all cases.

Other hyperparameters, such as the number of radial spheres and number of sphere levels in the CSGNN, matched those used in [12].

Dropout probability was set to 0.1. Dropout was applied both to intra- and inter-sphere convolution and in the hidden output layers.

The optimal hyperparameters for the three training sets are shown in Table 4-1. The training history for the split-temperature model is shown in Figure 4-11. Because the low- and high- temperature training sets were not the primary focus of this work, their training histories are omitted for brevity.

Table 4-1. Optimal hyperparameters and resulting number of unknowns for the three training sets.

Training Set	Hidden Output Layers	Parameter Factor	Learning Rate	Unknowns
Split Temperature	6	10	0.1	6.22e7
Low Temperature	6	8	0.1	3.99e7
High Temperature	3	12	0.1	6.11e7

4.3.3.3. Inference

Figure 4-12 shows the DFT-computed band energy (that is, the "truth") for all the snapshots in the training and test sets. A few features are noteworthy. First, for the solids, band energy decreases monotonically with temperature. Ideally, a model trained on snapshots at multiple temperatures will reproduce this trend. Second, liquids have substantially lower band energy than solids, even when they have the same temperature, as occurs at 933K. This illustrates what we pointed out earlier, that liquid snapshots are different from solids.

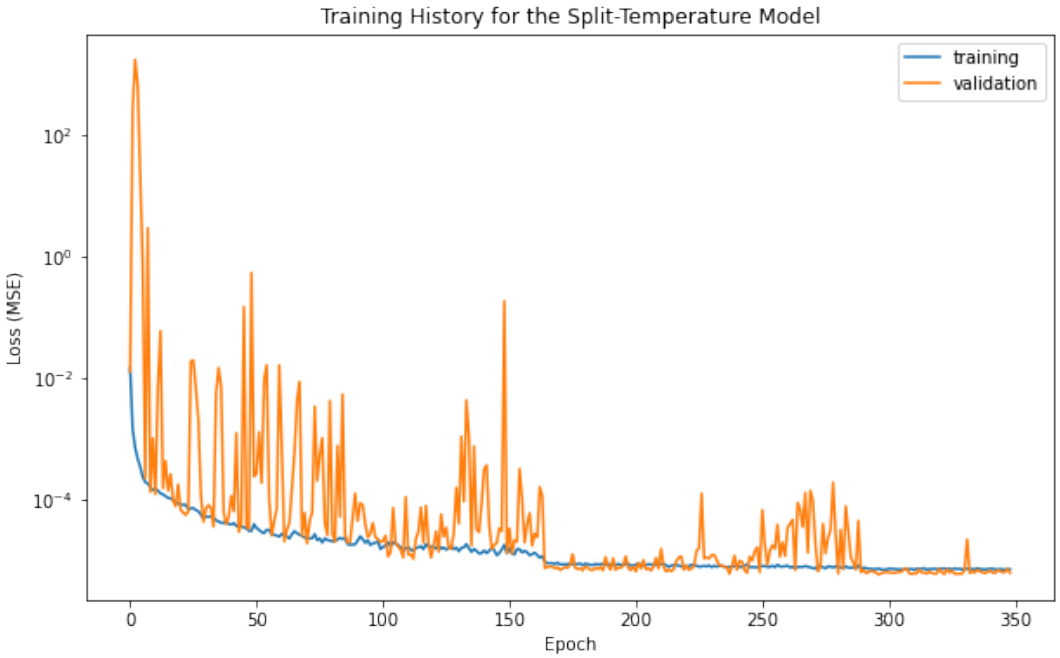


Figure 4-11. Training and validation errors during training for the split-temperature model.

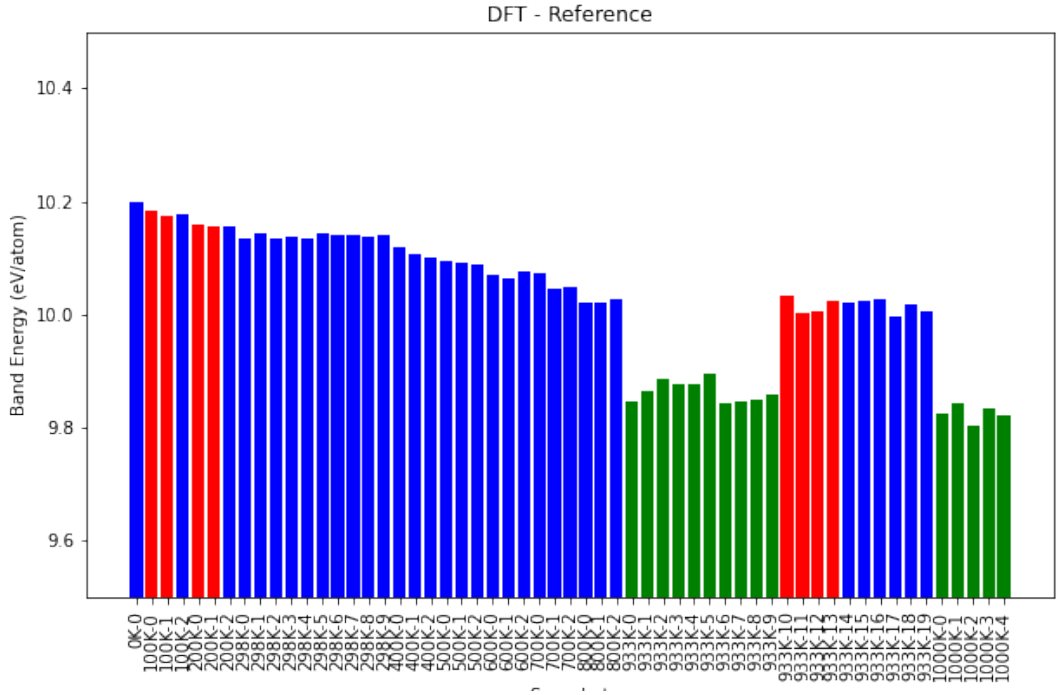


Figure 4-12. DFT computed reference band energies. The red bars are the solid training set snapshots. Blue are solid test snapshots, and green are liquid test snapshots.

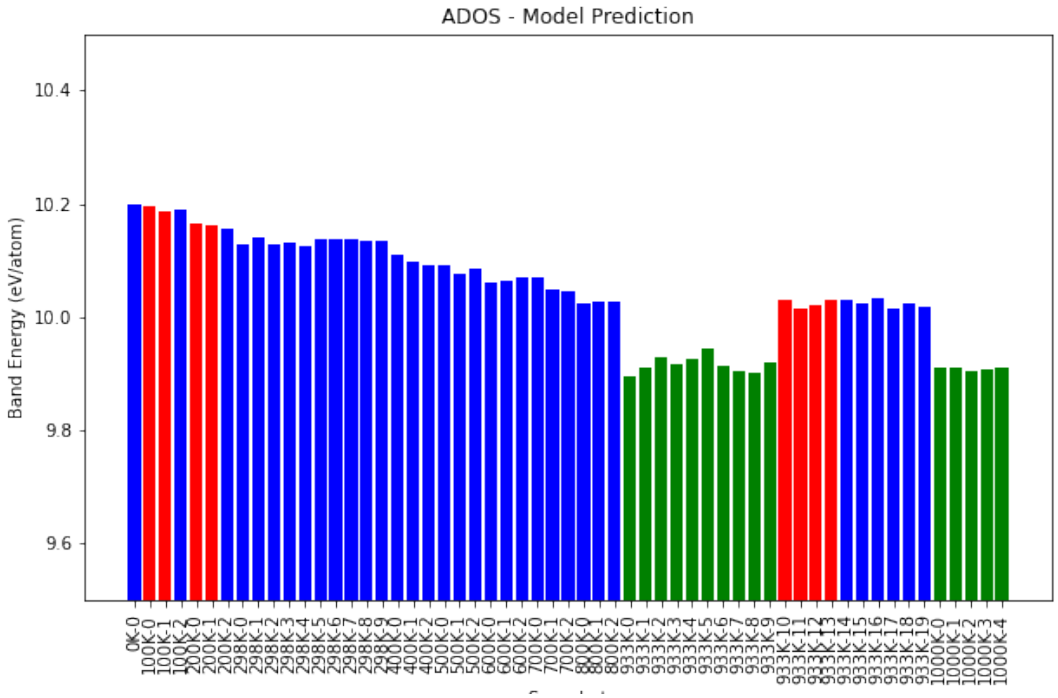


Figure 4-13. Band energies predicted by the split-temperature ADOS model. The red bars are the solid training set snapshots. Blue are solid test snapshots, and green are liquid test snapshots.

The split temperature model was used to predict ADOS (band energy, ultimately) for all snapshots. To produce these results, inference was performed on the model without dropout. The predictions are shown in Figure 4-13

It can immediately be seen that the model predictions qualitatively capture the two features we noted above—the temperature trend and the difference between solids and liquids—and that it is able to do so even though no liquids were present in the training set.

The parity plot in Figure 4-14 compares the DFT reference and ADOS model predictions directly. Points at higher energy are for lower temperatures, and vice versa. For all the solid snapshots, including those at intermediate temperatures, which were not included in the training set, very good agreement is obtained. The dashed lines on the plot are spaced at "chemical accuracy", ± 10 meV/atom, and predictions for solids are typically within this band. This result demonstrates that it is possible to generate machine learned models using the ADOS approach that exhibit at least some degree of transferability outside of the training set, which is an important property for practical use. Although predictions for the liquid snapshots are well outside the chemical accuracy window, they are not completely unphysical.

Figure 4-15 shows parity plots for models generated from the low-temperature and high-temperature training sets. These suggest that training on a single (or narrow range) of temperatures is inadequate to produce models that show the same level of transferability that the split-temperature model achieved.

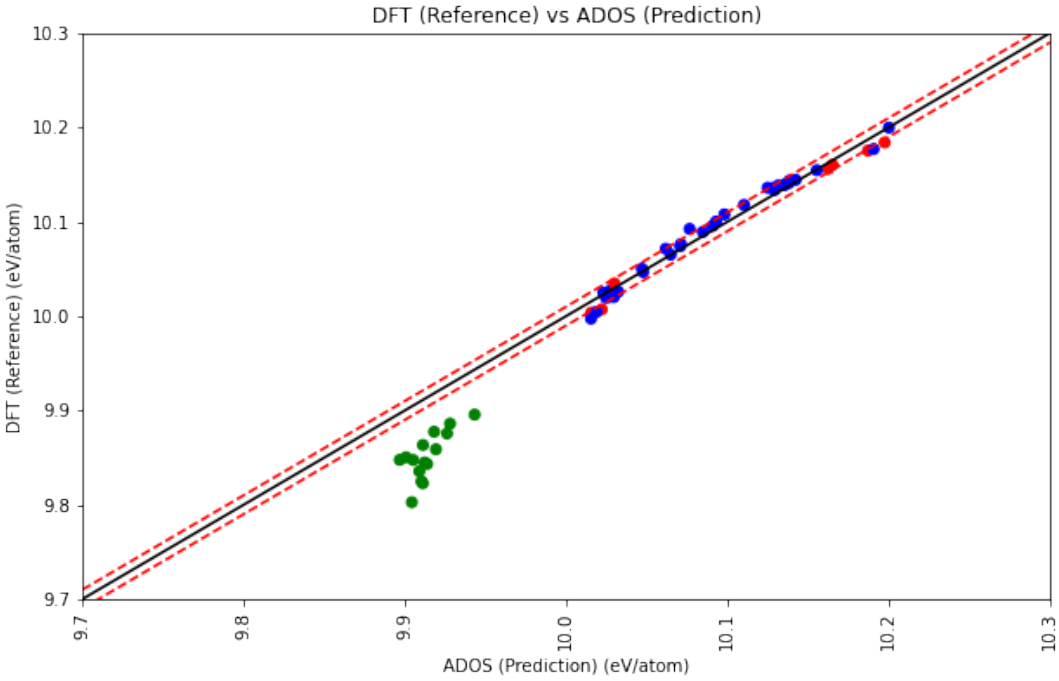
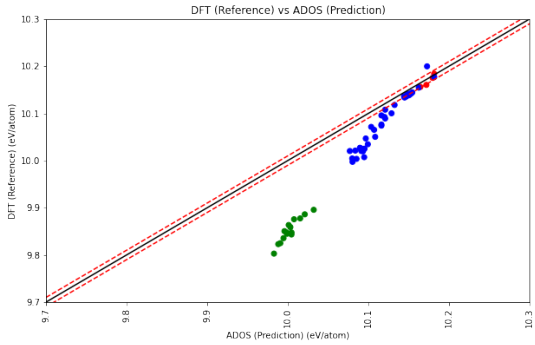
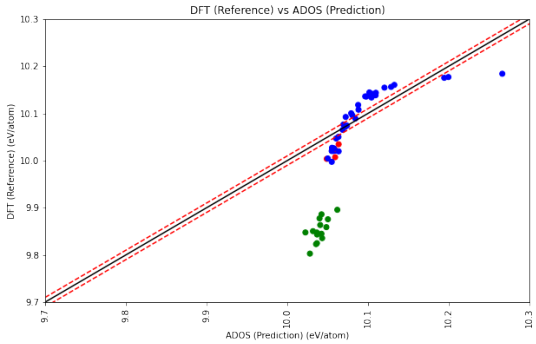


Figure 4-14. DFT versus split-temperature ADOS prediction of snapshot band energy. Red: solid training snapshot; Blue: solid test snapshot; Green: liquid test snapshot. The dash lines are ± 10 meV/atom above and below the center line.



(a) Low Temperature



(b) High Temperature

Figure 4-15. Parity plots for the low temperature and high temperature models. Red: solid training snapshot; Blue: solid test snapshot; Green: liquid test snapshot. The dash lines are $\pm$ 10 meV/atom above and below the center line.

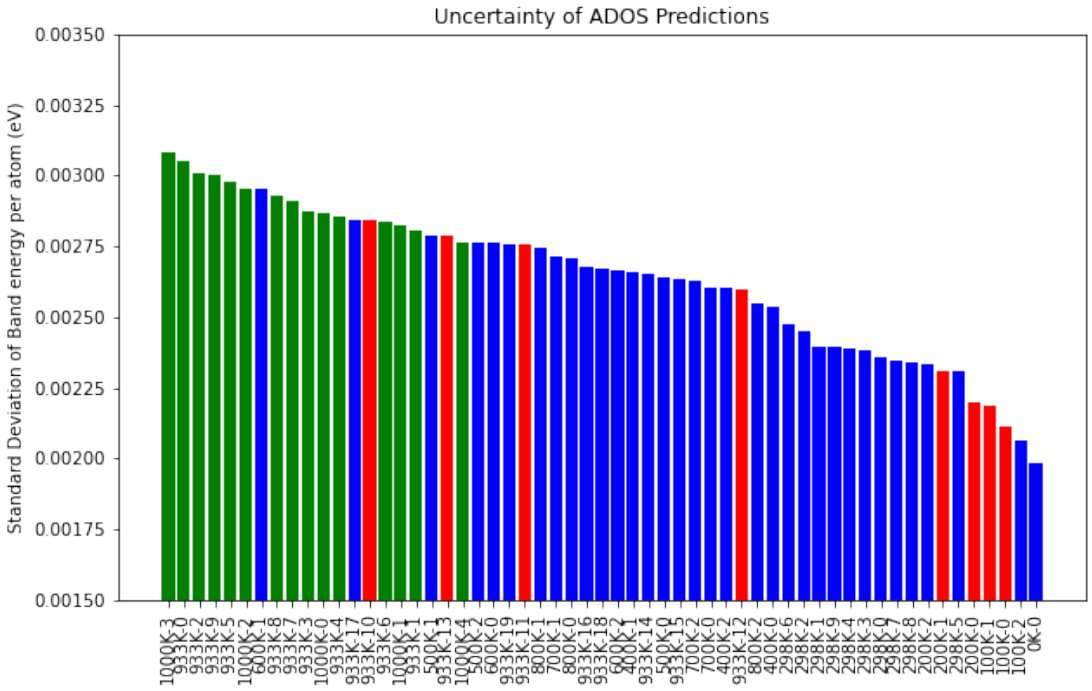


Figure 4-16. Standard deviation of dropout predictions for the split-temperature model. The snapshots have been sorted by decreasing uncertainty.

4.3.3.4. Dropout Uncertainty

Monte Carlo dropout was used to compute the uncertainty of the split-temperature model for all available snapshots. As in the training, a dropout probability of 0.1 was used. A total of 512 inferences were run, and the standard deviation of the resulting band energies predictions was calculated. The results are shown in Figure 4-16.

Notably, the dropout uncertainty of many of the test snapshots are lower than that of snapshots that were present in the training set. This is perhaps not too surprising, considering how well the model predicts the band energies of all the solid snapshots.

We also observe that liquid snapshots exhibit, for the most part, the greatest uncertainty. This is as we expected. By passing this basic test, dropout uncertainty is not ruled out as a possible approach to experimental design.

There is no discernible step change between solid and liquid uncertainty, which may be surprising, considering the large difference between their band energies. We again point to the success of the model in predicting the band energy of the liquids as a possible explanation.

4.3.4. Conclusions about Monte Carlo Dropout

Uncertainty predicted using Monte Carlo dropout provides a plausible way of selecting snapshots with which to augment an existing training set. It correctly indicates that test snapshots that intuitively are most different from the training set and that in fact have the greatest error should be included. We unfortunately were unable to develop a full strategy for making use of this information due to time limitations, but the results presented here suggest some obvious directions for possible future work. Moreover, the transferability of the split-temperature model to intermediate temperatures is an encouraging result that points to the broader applicability of the ADOS approach.

4.4. Conclusions and Future Work in Experimental Design

Under the current approach of mapping a dense spatial grid of fingerprints to LDOS, MC dropout is very costly. Thus, we started our investigation of dropout using atom-centered descriptors where inference becomes much less expensive. The initial results presented above on the ADOS approach indicate that Monte Carlo dropout is a promising direction for selecting snapshots with which to augment an existing training set.

There are other approaches that could be investigated as well. Deep ensembles have the advantage of having separately trained neural networks to use for performing prediction uncertainty. These may provide a better estimate of total prediction uncertainty, especially if they can capture differences in potential architectures and initial training points. However, deep ensembles require significant training neural network training iterations. This can be very costly given that one hyper-parameter training optimization for an LDOS model may take days. The cost of performing the various ML training runs necessary to assess uncertainty in predictions is a limitation when utilizing MC dropout or deep ensembles to select candidates in an adaptive design framework. Transfer learning is another tool that could provide more insight about the applicability of various data sets to use for augmentation. We briefly discuss results to date on transferability below.

4.4.1. Transfer Learning

Although it is not the same as OED, transfer learning has some related themes. Transfer learning is a class of machine learning approaches that are designed to train models that perform well in circumstances where the model application is somewhat different from the training application.[32] Kyle Lennon, a CSRI intern in the summer of 2021, investigated transfer learning for the ML-DFT application. He used a transfer learning approach called Deep Variational Information Bottleneck neural network (Deep VIB) to learn on one training data set (e.g. 298K solid) with the goal of extrapolating well to another data set (e.g. 933K solid).[28]

Unfortunately, his results were inconclusive: the transfer learning approach did not show good transferability. One hypothesis from the study was that more snapshot data (from different configurations) would be needed to train these types of Deep VIB networks to work well. Another conclusion was that the 91-dimensional bispectrum input contained redundant information: the Deep VIB identified a space

of 32 latent features, a significant reduction in input dimension. This finding is similar to the PCA findings. Future work might involve looking at latent features in the Deep VIB network as part of the OED process.

5. SOFTWARE

The software utilized in portions of this project is publicly available as the Materials Learning Algorithms (MALA), available repository on GitHub at https://github.com/mala-project/mala. The software is a machine learning code base written in python which calls and utilizes specific functionalities of both LAMMPS and Quantum Espresso(QE), open source atomistic modeling software packages. The use of LAMMPS and QE were chosen to provide opportunities for customization and editing for this specific project's use cases. The installation and usage of the MALA package will be dependent to each individual machine's configuration and software environment.

A general outline of the installation process is provided on the MALA repository site as a starting point to be able to successfully use the MALA software. At Sandia the MALA software package has been installed on both Blake and HPC production clusters. This was not a trivial task as the machines have varying software environments, hardware architectures, technical support resources, and permissions. The following workflow has been tested and successfully used to build and run the MALA package on production machines.

5.1. Machine setup

The installation and use of MALA, LAMMPS, and Quantum Espresso are computationally demanding programs. Selection of the software environment, calculation routines, and mathematical libraries is important to maximize the efficiency of the overall software package and minimize the computational resources needed for each calculation. The software environment is set up on our machines using GNU compilers, CMake, and openmpi parallelization. To increase the speed of calculations in Quantum Espresso MKL libraries and FFTW3 are used.

With the machine environment set there are five sections to the installation of MALA. 1) Installation and build of LAMMPS 2) Installation and build of Quantum Espresso 3) Downloading MALA 4) Installation and build of Conda environment. 5) Installation of MALA + linking with LAMMPS and Quantum Espresso.

5.2. Installation and build of LAMMPS

The specific LAMMPS build that is utilized is found at https://github.com/athomps/lammps.git. After the download is complete the compute-grid-new branch is utilized. The use of LAMMPS in MALA is specific to the use of the ML-SNAP package, which is used to derive accurate potentials fit from DFT data. The build of LAMMPS includes only the ML-SNAP package and is built to have a shared library for use with mpi. If no errors are shown during the build process LAMMPS has been successfully built.

5.3. Installation and build of Quantum Espresso

The MALA package uses a specific version of Quantum Espresso (v6.4.1) which can be downloaded and installed from the QEF GitHub repository. The specific version of Quantum Espresso can be accessed by going into the q-e/ directory and checking out the qe-6.4.1 branch. After correctly accessing v6.4.1 the build procedure outlined for Quantum Espresso can be followed.
For use of the total energy module in MALA a new external directory must be copied into the q-e head directory. This total energy module directory can be located at https://gitlab.com/casus/q-e/-/tree/tem_original_development.

5.4. Download MALA

The MALA package must be downloaded as it contains build files for setting up a Conda environment. This can be done by cloning the repo at https://github.com/mala-project/mala.git.

5.5. Installation and build of Conda environment

In the /mala/install/ directory are specific .yml files which can be used to build a Conda environment specific to MALA. To be able to use the build files a Conda module must be loaded or a Conda program installed for user use. This will solve an environment to accomodate all listed build packages and install them into an environment. The new environment can now be activated and used to run either run MALA or continue with further installation steps of specific external modules. For packages that cannot be installed via Conda, they can be found and installed via Pip into the conda environment. Below is a list of a current working Conda environment for MALA which utilizes LAMMPS and Quantum espresso.

5.6. Installing MALA and linking to LAMMPS and Quantum Espresso

Following the activation of the Conda environment the MALA module can be installed in the head MALA directory. The LAMMPS build can be linked to the Conda environment by returning to the LAMMPS/src/ directory and installing the python link by *make install-python*. The Quantum espresso total energy module can be built by returning to the q-e/total_energy_module/ directory and executing the built_total_energy_module.sh file.

Machine Modules:
gnu/7.3.1, openmpi-gnu/4.1, mkl/20.0.2.254, cmake/3.20.3

LAMMPS 2022.6.2
Quantum Espresso v6.4.1

Conda Environment:
Conda pacakges: nomkl, python>=3.8, pip, numpy, scipy, optuna, ase, mpmath, tensorboard, pandas, swig, libjpeg-turbo

Pip packages: oapackage, tensorboard, mpi4py, pytest, torch, torchaudio, torchvision, tensorflow-cpu

6. CONCLUSION

This project developed a new surrogate model for density functional theory using deep neural networks. The developed ML surrogate is demonstrated in a workflow to generate accurate band energies, total energies, and density of the 298K and 933K Aluminum systems. Furthermore, the models can be used to predict the quantities of interest for systems with more number of atoms than the training data set. We have demonstrated that the ML model can be used to compute the quantities of interest for systems with 100,000 Al atoms. When compared with 2000 Al system the new surrogate model is as accurate as DFT, but three orders of magnitude faster. We also explored optimal experimental design techniques to choose the training data and novel Graph Neural Networks to train on smaller data sets. These are promising methods that need to be explored in the future.

BIBLIOGRAPHY

[1] Nist/sematech e-handbook of statistical methods, eds. c.croarkin and p. tobias, 2020.

[2] Albert P Bartók, Mike C Payne, Risi Kondor, and Gábor Csányi. Gaussian approximation potentials: The accuracy of quantum mechanics, without the electrons. *Physical review letters*, 104(13):136403, 2010.

[3] Jörg Behler and Michele Parrinello. Generalized neural-network representation of high-dimensional potential-energy surfaces. *Physical review letters*, 98(14):146401, 2007.

[4] Felix Brockherde, Leslie Vogt, Li Li, Mark E Tuckerman, Kieron Burke, and Klaus-Robert Müller. Bypassing the kohn-sham equations with machine learning. *Nature communications*, 8(1):1–10, 2017.

[5] Kenneth P Burnham and David R Anderson. *Model selection and multimodel inference: a practical information-theoretic approach*. Springer Science & Business Media, 2003.

[6] Kathryn Chaloner and Isabella Verdinelli. Bayesian experimental design: A review. *Statist. Sci.*, 10(3):273–304, 08 1995.

[7] Anand Chandrasekaran, Deepak Kamal, Rohit Batra, Chiho Kim, Lihua Chen, and Rampi Ramprasad. Solving the electronic structure problem with machine learning. *npj Computational Materials*, 5(1):1–7, 2019.

[8] R Drautz. Atomic cluster expansion for accurate and transferable interatomic potentials. *Physical Review B*, 99(1):014104, 2019.

[9] J. A. Ellis, L. Fiedler, G. A. Popoola, N. A. Modine, J. A. Stephens, A. P. Thompson, A. Cangi, and S. Rajamanickam. Accelerating finite-temperature kohn-sham density functional theory with deep neural networks. *Phys. Rev. B*, 104:035120, Jul 2021.

[10] J. Austin Ellis, Attila Cangi, Normand A. Modine, J. Adam Stephens, Aidan P. Thompson, and Sivasankaran Rajamanickam. Accelerating finite-temperature kohn-sham density functional theory with deep neural networks, 2020.

[11] Adam Foster, Martin Jankowiak, Elias Bingham, Paul Horsfall, Yee Whye Teh, Thomas Rainforth, and Noah Goodman. Variational bayesian optimal experimental design. In *Advances in Neural Information Processing Systems*, pages 14036–14047, 2019.

[12] James Fox, Normand A Modine, and Sivasankaran Rajamanickam. Accelerating electronic structure calculation with atom-decomposed neural modeling. CSRI Summer Proceedings 2021, 2021.

[13] James Fox, Bo Zhao, Sivasankaran Rajamanickam, Rampi Ramprasad, and Le Song. Concentric spherical gnn for 3d representation learning, 2021.

[14] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 1050–1059. JMLR.org, 2016.

[15] Paolo Giannozzi, Stefano Baroni, Nicola Bonini, Matteo Calandra, Roberto Car, Carlo Cavazzoni, Davide Ceresoli, Guido L. Chiarotti, Matteo Cococcioni, Ismaila Dabo, Andrea Dal Corso, Stefano de Gironcoli, Stefano Fabris, Guido Fratesi, Ralph Gebauer, Uwe Gerstmann, Christos Gougoussis, Anton Kokalj, Michele Lazzeri, Layla Martin-Samos, Nicola Marzari, Francesco Mauri, Riccardo Mazzarello, Stefano Paolini, Alfredo Pasquarello, Lorenzo Paulatto, Carlo Sbraccia, Sandro Scandolo, Gabriele Sclauzero, Ari P. Seitsonen, Alexander Smogunov, Paolo Umari, and Renata M. Wentzcovitch. QUANTUM ESPRESSO: A modular and open-source software project for quantum simulations of materials. *Journal of Physics: Condensed Matter*, 21(39):395502, September 2009.

[16] Paolo Giannozzi, Oscar Baseggio, Pietro Bonfà, Davide Brunato, Roberto Car, Ivan Carnimeo, Carlo Cavazzoni, Stefano de Gironcoli, Pietro Delugas, Fabrizio Ferrari Ruffino, Andrea Ferretti, Nicola Marzari, Iurii Timrov, Andrea Urru, and Stefano Baroni. Quantum ESPRESSO toward the exascale. *The Journal of Chemical Physics*, 152(15):154105, April 2020.

[17] W R Gilks, S Richardson, and D J Spiegelhalter. *Markov Chain Monte Carlo in Practice*. Chapman & Hall, 1996.

[18] D. R. Hamann. Optimized norm-conserving Vanderbilt pseudopotentials. *Physical Review B*, 88(8):085117, August 2013.

[19] Katja Hansen, Franziska Biegler, Raghunathan Ramakrishnan, Wiktor Pronobis, O Anatole Von Lilienfeld, Klaus-Robert Muller, and Alexandre Tkatchenko. Machine learning predictions of molecular properties: Accurate many-body potentials and nonlocality in chemical space. *The journal of physical chemistry letters*, 6(12):2326–2331, 2015.

[20] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580, 2012. cite arxiv:1207.0580.

[21] Xun Huan and Youssef M Marzouk. Sequential bayesian optimal experimental design via approximate dynamic programming. *arXiv preprint arXiv:1604.08320*, 2016.

[22] Ian T Jolliffe. *Principal component analysis, second edition*. Springer Series in Statistics, 2002.

[23] Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998.

[24] MR Khodja, MD Prange, and HA Djikpesse. Guided bayesian optimal experimental design. *Inverse Problems*, 26(5):055008, 2010.

[25] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *International Conference on Learning Representations*, 2017.

[26] Walter Kohn. Density functional and density matrix method scaling linearly with the number of atoms. *Physical Review Letters*, 76(17):3168, 1996.

[27] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6405–6416, Red Hook, NY, USA, 2017. Curran Associates Inc.

[28] Kyle Lennon and Sivasankaran Rajamanickam. Learning transferable neural network surrogates for kohn-sham density functional theory. In *2021 Summer Proceedings of the Computer Science Research Institute, Sandia National Laboratories*, 2021.

[29] Li Li, John C Snyder, Isabelle M Pelaschier, Jessica Huang, Uma-Naresh Niranjan, Paul Duncan, Matthias Rupp, Klaus-Robert Müller, and Kieron Burke. Understanding machine-learned density functionals. *International Journal of Quantum Chemistry*, 116(11):819–833, 2016.

[30] Hendrik J. Monkhorst and James D. Pack. Special points for Brillouin-zone integrations. *Physical Review B*, 13(12):5188–5192, June 1976.

[31] Douglas Montgomery. *Design and Analysis of Experiments, 10th ed.* Wiley, 2019.

[32] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[33] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[34] John P. Perdew, Adrienn Ruzsinszky, Gábor I. Csonka, Oleg A. Vydrov, Gustavo E. Scuseria, Lucian A. Constantin, Xiaolan Zhou, and Kieron Burke. Restoring the Density-Gradient Expansion for Exchange in Solids and Surfaces. *Physical Review Letters*, 100(13):136406, April 2008.

[35] Steve Plimpton. Fast parallel algorithms for short-range molecular dynamics. *Journal of computational physics*, 117(1):1–19, 1995.

[36] L. Rueshendorff. Wasserstein metric. In *Encyclopedia of Mathematics*, 2011.

[37] Matthias Rupp, Alexandre Tkatchenko, Klaus-Robert Müller, and O Anatole Von Lilienfeld. Fast and accurate modeling of molecular atomization energies with machine learning. *Physical review letters*, 108(5):058301, 2012.

[38] Matthias Rupp, Alexandre Tkatchenko, Klaus-Robert Müller, and O. Anatole von Lilienfeld. Fast and accurate modeling of molecular atomization energies with machine learning. *Phys. Rev. Lett.*, 108:058301, Jan 2012.

[39] Jerome Sacks, William J Welch, Toby J Mitchell, and Henry P Wynn. Design and analysis of computer experiments. *Statistical Science*, pages 409–423, 1989.

[40] Thomas J Santner, Brian J Williams, and William I Notz. *The design and analysis of computer experiments*. Springer Series in Statistics, 2003.

[41] Kristof T Schütt, Farhad Arbabzadah, Stefan Chmiela, Klaus R Müller, and Alexandre Tkatchenko. Quantum-chemical insights from deep tensor neural networks. *Nature communications*, 8(1):1–8, 2017.

[42] Kristof T Schütt, Henning Glawe, Felix Brockherde, Antonio Sanna, Klaus-Robert Müller, and Eberhard KU Gross. How to represent crystal structures for machine learning: Towards fast prediction of electronic properties. *Physical Review B*, 89(20):205118, 2014.

[43] Timothy Simpson, Vasilli Toropov, Vladimir Balabanov, and Felipe Viana. Design and analysis of computer experiments in multidisciplinary design optimization: A review of how far we have come-or not. In *12th AIAA/ISSMO multidisciplinary analysis and optimization conference*, page 5802, 2008.

[44] John C Snyder, Matthias Rupp, Katja Hansen, Leo Blooston, Klaus-Robert Müller, and Kieron Burke. Orbital-free bond breaking via machine learning. *The Journal of chemical physics*, 139(22):224104, 2013.

[45] John C Snyder, Matthias Rupp, Katja Hansen, Klaus-Robert Müller, and Kieron Burke. Finding density functionals with machine learning. *Physical review letters*, 108(25):253002, 2012.

[46] Aidan P Thompson, Laura P Swiler, Christian R Trott, Stephen M Foiles, and Garritt J Tucker. Spectral neighbor analysis method for automated generation of quantum-accurate interatomic potentials. *Journal of Computational Physics*, 285:316–330, 2015.

[47] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[48] Philippe F Weck, Kyle R Cochrane, Seth Root, J Matthew D Lane, Luke Shulenburger, John H Carpenter, Travis Sjostrom, Thomas R Mattsson, and Tracy J Vogler. Shock compression of strongly correlated oxides: A liquid-regime equation of state for cerium (iv) oxide. *Physical Review B*, 97(12):125106, 2018.

[49] M. A. Wood, M. A. Cusentino, B. D. Wirth, and A. P. Thompson. Data-driven material models for atomistic simulation. *Phys. Rev. B*, 99:184305, 2019.

[50] Qin Yang, Chenglin Li, Wenrui Dai, Junni Zou, Guo-Jun Qi, and Hongkai Xiong. Rotation equivariant graph convolutional network for spherical image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4303–4312, 2020.

[51] Kun Yao and John Parkhill. Kinetic energy of hydrocarbons as a function of electron density and convolutional neural networks. *Journal of chemical theory and computation*, 12(3):1139–1147, 2016.

DISTRIBUTION

Email—Internal

Name	Org.	Sandia Email Address
Technical Library	1911	sanddocs@sandia.gov

Hardcopy—Internal

Number of Copies	Name	Org.	Mailstop

Hardcopy—External

Number of Copies	Name(s)	Company Name and Company Mailing Address
1		

Sandia National Laboratories

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.