PNNL-38353

Energy-efficient, Large-scale Molecular Dynamics Simulations via Hardware- and Algorithm-level Optimization

September 2025

Xinyi Shen
Bruno Jacob
Anne Chaka
Yunxiang Chen
Amanda Howard
Elias Nakouzi
Sayan Ghosh
Sebastien Kerisit

Prepared for the U.S. Department of Energy
under Contract DE-AC05-76RL01830

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor Battelle Memorial Institute, nor any of their employees, makes **any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights**. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or Battelle Memorial Institute. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

PACIFIC NORTHWEST NATIONAL LABORATORY
operated by
BATTELLE
for the
UNITED STATES DEPARTMENT OF ENERGY
under Contract DE-AC05-76RL01830

Printed in the United States of America

Available to DOE and DOE contractors from
the Office of Scientific and Technical Information,
P.O. Box 62, Oak Ridge, TN 37831-0062
www.osti.gov
ph: (865) 576-8401
fox: (865) 576-5728
email: reports@osti.gov

Available to the public from the National Technical Information Service
5301 Shawnee Rd., Alexandria, VA 22312
ph: (800) 553-NTIS (6847)
or (703) 605-6000
email: info@ntis.gov
Online ordering: http://www.ntis.gov

PNNL-38353

Energy-efficient, Large-scale Molecular Dynamics Simulations via Hardware- and Algorithm-level Optimization

September 2025

Xinyi Shen
Bruno Jacob
Anne Chaka
Yunxiang Chen
Amanda Howard
Elias Nakouzi
Sayan Ghosh
Sebastien Kerisit

Prepared for
the U.S. Department of Energy
under Contract DE-AC05-76RL01830

Pacific Northwest National Laboratory
Richland, Washington 99354

Abstract

This work aims to develop a framework for energy-efficient computing that will enable molecular dynamics (MD) simulations of large-scale phenomena with atomic precision and simultaneously remove computational bottlenecks limiting the speed of MD simulations. We seek to implement such an approach through the development of surrogate models for the interatomic force calculation combined with the use of mixed numerical precision formats. For a model system of neutral atoms (only pairwise interactions), significant force calculation efficiency improvements were achieved, without detrimental effects on atomic structures or average energies, using single precision, by developing a surrogate model (deep neural network), and by quantizing this surrogate model. For a model system of charged atoms, the reciprocal-space calculation of electrostatic interactions was identified as the main bottleneck, and the development of a surrogate model should be pursued to achieve an estimated one-order-of-magnitude additional speedup.

Summary

The growth of supercomputing has led to the prevalent use of large-scale molecular dynamics (MD) simulations in many scientific domains. Despite efforts to develop rare-event techniques that bias MD algorithms towards physically and chemically interesting events, large-scale MD simulations are generally aimed at direct sampling of the relevant phase space and attempt to bridge molecular and particle scales only through brute force. This approach is highly energy inefficient, and a novel approach is thus needed to reduce the energy cost of computation and simultaneously remove bottlenecks that limit the speed of MD simulations.

The work presented in this report seeks to develop a framework for energy-efficient computing that will enable MD simulations of large-scale phenomena with atomic precision. Two approaches for energy-efficient computing were investigated: (1) Performance optimization through use of mixed numerical precision formats; and (2) Force calculation optimization through surrogate model development.

Performance optimization through use of mixed precision formats employed two model systems: argon (Ar) and sodium chloride (NaCl). Simulations of neutral Ar atoms isolated the pairwise short-range interaction calculation, which was greatly accelerated on GPUs compared to CPUs, such that the pairwise calculation ceased to be the computing bottleneck. Lowering the precision format led to faster calculations without any loss of accuracy in terms of calculated structures. Simulations of ionic NaCl require calculations of both short-range and long-range/electrostatic interactions, where part of the latter is performed in reciprocal space (k-space). Significant speedup with no loss of accuracy was achieved by using NVIDIA's GPU-accelerated fast Fourier transform library for the k-space calculation. However, unlike for the Ar system, efficiency gains from lowering the precision format only became apparent for large systems (>250,000 atoms). Nonetheless, the k-space calculation remained the main bottleneck, indicating that it should be the focus of future efficiency gain efforts.

A deep neural network (DNN) surrogate model was developed for the force calculation in liquid Ar simulations. The force model is a compact pairwise multilayer perceptron with widths [1, 64, 64, 64, 1] and tanh activations in the hidden layers. It uses the normalized interatomic distance as input and outputs a scalar weight applied along the unit displacement vector. Structural and energy fidelity of the surrogate model to the MD simulation was demonstrated. Quantization experiments were run using BF16 and INT4. BF16 reduced the force-evaluation compute time by ~25–27% with no effect on predicted structure and energy, while the use of INT4 resulted in a clear deterioration of both structure and energy. Neighbor search timings were largely insensitive to precision and dominated end-to-end iteration time.

Future work should investigate scaling of numerical precision effects with larger and more complex molecular systems to avoid latency and tail effects. The DNN surrogate model should be extended to more complex systems that include electrostatic and multibody interactions. Based on our performance evaluation with GPUs and the assumption that the force-evaluation compute time of the DNN is independent of the complexity of the MD forcefield it is replacing, we are predicting that replacing the k-space calculation by a DNN surrogate model will lead to a one-order-of-magnitude speedup. The DNN should eventually be embedded in MD software like LAMMPS to benefit from its efficient neighbor search and other features.

Acknowledgments

This research was supported by the Physical & Computational Sciences Directorate Mission Seed Investment, under the Laboratory Directed Research and Development (LDRD) Program at Pacific Northwest National Laboratory (PNNL). PNNL is a multi-program national laboratory operated for the U.S. Department of Energy (DOE) by Battelle Memorial Institute under Contract No. DE-AC05-76RL01830.

Acronyms and Abbreviations

AIMD	Ab initio molecular dynamics
CPU	Central processing unit
DNN	Deep neural network
DOE	Department of Energy
GNN	Graph neural network
GPU	Graphics processing unit
LAMMPS	Large-scale atomic/molecular massively parallel simulator
LDRD	Laboratory Directed Research and Development
MD	Molecular dynamics
ML	Machine learning
MPS	Multi-process service
MSE	Mean square error
NERSC	National Energy Research Scientific Computing Center
PCSD	Physical and Computational Sciences Directorate
PNNL	Pacific Northwest National Laboratory
RDF	Radial distribution function
RMSE	Root mean square error

Contents

Figures

1.0 Introduction

Traditional molecular dynamics (MD) simulations have exclusively employed double-precision floating-point arithmetic for numerical stability and accuracy on contemporary graphics processing units (GPUs). Existing studies (e.g., Le Grand et al. (2013)) of GPU-enabled MD simulations primarily discussed adaptations for single-precision. Recently (Jia et al., 2020), machine learning (ML) based MD schemes have employed mixed single and half precision for training on the latest GPUs (which support a variety of mixed-precision formats at the hardware level), improving the overall performance/watt and energy consumption (owing to reduced data movement and higher throughput). Automatic mixed-precision and post-training quantization (to exploit integer precision) can offer major performance and energy-efficiency advantages of ML models over full precision. We will specifically investigate the performance/accuracy trade-offs on MD datasets using various mixed-precision optimizations.

Such hardware-level optimization can be augmented by algorithm-level optimization, specifically using ML techniques. ML techniques have been applied to MD simulations to address challenges and limitations with inadequate phase space sampling (Trizio and Parrinello, 2021), low accuracy of forcefields (Behler, 2021), and analysis of atomic trajectories (Plante et al., 2019). In this work, algorithm-level optimization will be achieved using surrogate modeling with deep learning. Once trained, these models reduce the cost of complex pairwise and electrostatic interactions used in the force computation.

The approach described in this report will be applicable to any science domain where simulations of large-scale phenomena require atomistic precision. A key example is solid-fluid interfacial systems where atomic-level interactions at the interface have a profound effect on the large-scale behavior of the system. Examples abound in catalysis, materials synthesis, geochemistry, energy storage, etc. As such, this work could open the door to new scientific understanding by removing computational barriers that have historically limited spatial and temporal scales achievable with molecular-scale simulation.

1.1 Research Design and Methodology

We followed a two-pronged approach to enhance hardware and algorithm efficiency for MD workloads, by trading-off precision with computation efficiency (as depicted in Figure 1), relying on: 1) simulations exploiting mixed-precision computations (lower-precision formats consume less memory and increase bandwidth); and 2) surrogate modeling of interatomic forces using deep neural networks (DNNs) . MD simulations of liquid Ar and crystalline NaCl will serve as test cases where the former only requires calculation of short-range interactions whereas the latter requires calculation of both short-range and long-range electrostatic interactions. Overall, we use the MD trajectories (generated from mixed-precision simulations) to train a DNN surrogate model that approximates the interatomic forces, further leveraging post-training quantization to utilize contemporary lower-precision data formats such as brain floating point (i.e., BF16). Computing resources of Pacific Northwest National Laboratory (PNNL) Research Computing (i.e., Deception cluster) and National Energy Research Scientific Computing Center (NERSC) leadership computing facilities (i.e., Perlmutter supercomputer) were used in this research.

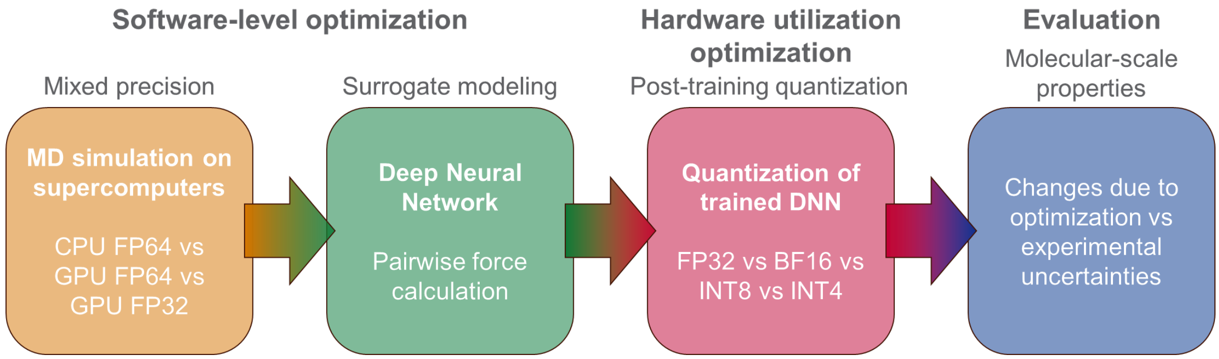
Software-level optimization
Hardware utilization optimization
Evaluation
Mixed precision
Surrogate modeling
Post-training quantization
Molecular-scale properties
MD simulation on supercomputers
CPU FP64 vs GPU FP64 vs GPU FP32
Deep Neural Network
Pairwise force calculation
Quantization of trained DNN
FP32 vs BF16 vs INT8 vs INT4
Changes due to optimization vs experimental uncertainties

Figure 1. The approach employed in this work consists of software-level optimization using mixed numerical precision formats and a surrogate model as well as hardware utilization optimization via post-training quantization of the surrogate model. Future work will evaluate how changes to predicted structural properties due to this optimization approach compare to the magnitude of experimental uncertainties.

1.2 Model Systems

The interaction energy between two atoms *i* and *j*, U_{ij}, in classical MD simulations is composed of two terms:

$$U_{ij} = \frac{1}{4\pi\varepsilon_0}\frac{q_i q_j}{r_{ij}} + \varphi(r_{ij}) \qquad \text{Eq. 1-1}$$

where ε_0 is the permittivity of vacuum, q_i and q_j are the charges on atoms *i* and *j* separated by distance r_{ij}, and $\varphi(r_{ij})$ is a function describing the short-range pairwise interactions between atoms *i* and *j*.

Two model systems (Figure 2) were considered:

1. Argon. In this system, $q = 0$ and Eq. 1-1 therefore simplifies to the short-range pairwise interaction only.

2. NaCl. In this system, $q = |1|$, and the interaction energy results from both long-range electrostatic interactions and short-range pairwise interactions.

We will investigate the effects of optimization on structural (e.g., radial distribution functions) and energetic (e.g., ensemble averages) properties of the two model systems.

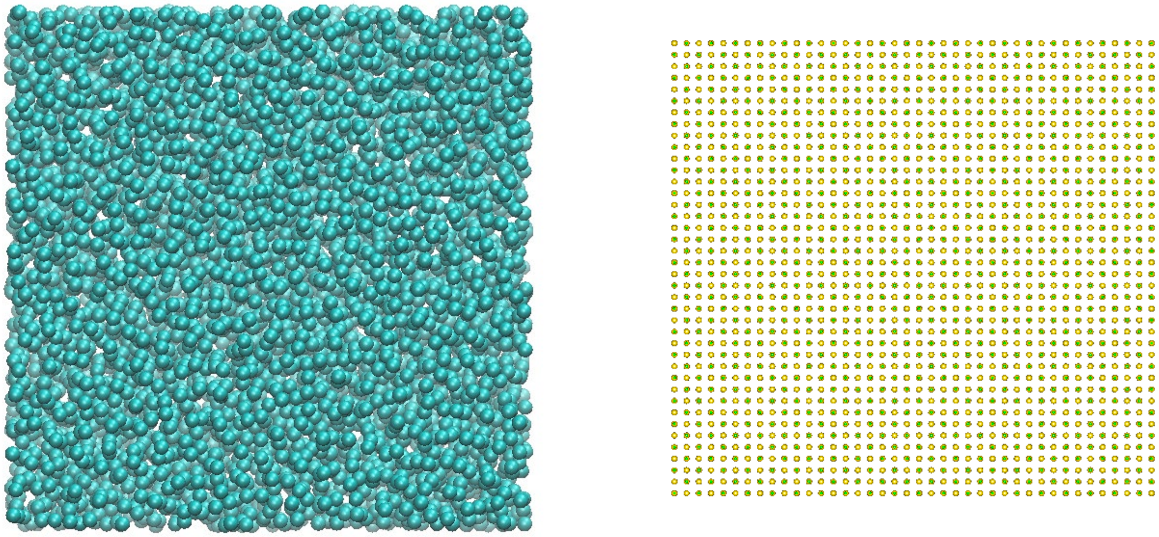

Figure 2. Snapshots of the model systems used in this work: Liquid Ar (left) and crystalline NaCl (right).

2.0 Software-level Optimization with Mixed Numerical Precision Formats

Two model systems were considered: Ar (Section 2.1) and NaCl (Section 2.2). Ar is a neutral monoatomic system that allows for isolating the pairwise interaction calculation, whereas simulating NaCl requires calculating electrostatic interactions, part of which is performed in reciprocal space (k-space).

2.1 Argon System: Pairwise Interactions

Extensive testing of the GPU package of LAMMPS and of Kokkos_fp32[1] was performed by varying the number of atoms in the system, the simulation ensemble, the number of GPUs used, and the computing platform all for double, mixed, and single numerical precision formats. Time per simulation step was used as the performance metric. The simulations performed for this evaluation were run at 60 K (i.e., for solid Ar).

For the simulation ensemble, simulations in the NVE ensemble (constant number of particles, constant volume, and constant energy) were faster than simulations in the NPT ensemble (constant number of particles, constant pressure, and constant temperature) as shown in Figure 3(a), consistent with the additional computational overhead associated with thermostat and barostat updates in the NPT ensemble. For both ensembles, reduced numerical precision led to increased performance, although the difference between mixed and single precision was relatively small for the NVE ensemble. Unless specified otherwise, the simulations presented hereafter in Section 2.1 were carried out in the NPT ensemble to enable a consistent performance evaluation.

Performance improved with the number of GPUs used up to 4 GPUs (Figure 3(b)). For 8 GPUs, performance fell at times below that of 4 GPUs for smaller systems but surpassed that for 4 GPUs once the number of atoms was sufficiently large. Reproducibility tests (Figure 4) showed that simulations performed using 1 or 2 GPUs were highly consistent, whereas simulations that used 4 GPUs exhibited slight variations at low system sizes and results with 8 GPUs displayed significant variability between repeated tests. The optimal GPU count was not necessarily the maximum available but depended on the system size. The low performance and reduced reproducibility at high GPU counts arose from increased communication and greater sensitivity to scheduling, which became critical when the per-GPU workload was small. The simulations presented hereafter were performed with a single GPU for consistency and ease of comparison.

[1] Kokkos_fp32 is an experimental extension of the Kokkos backend for LAMMPS aimed at enabling single- and mixed-precision simulations.

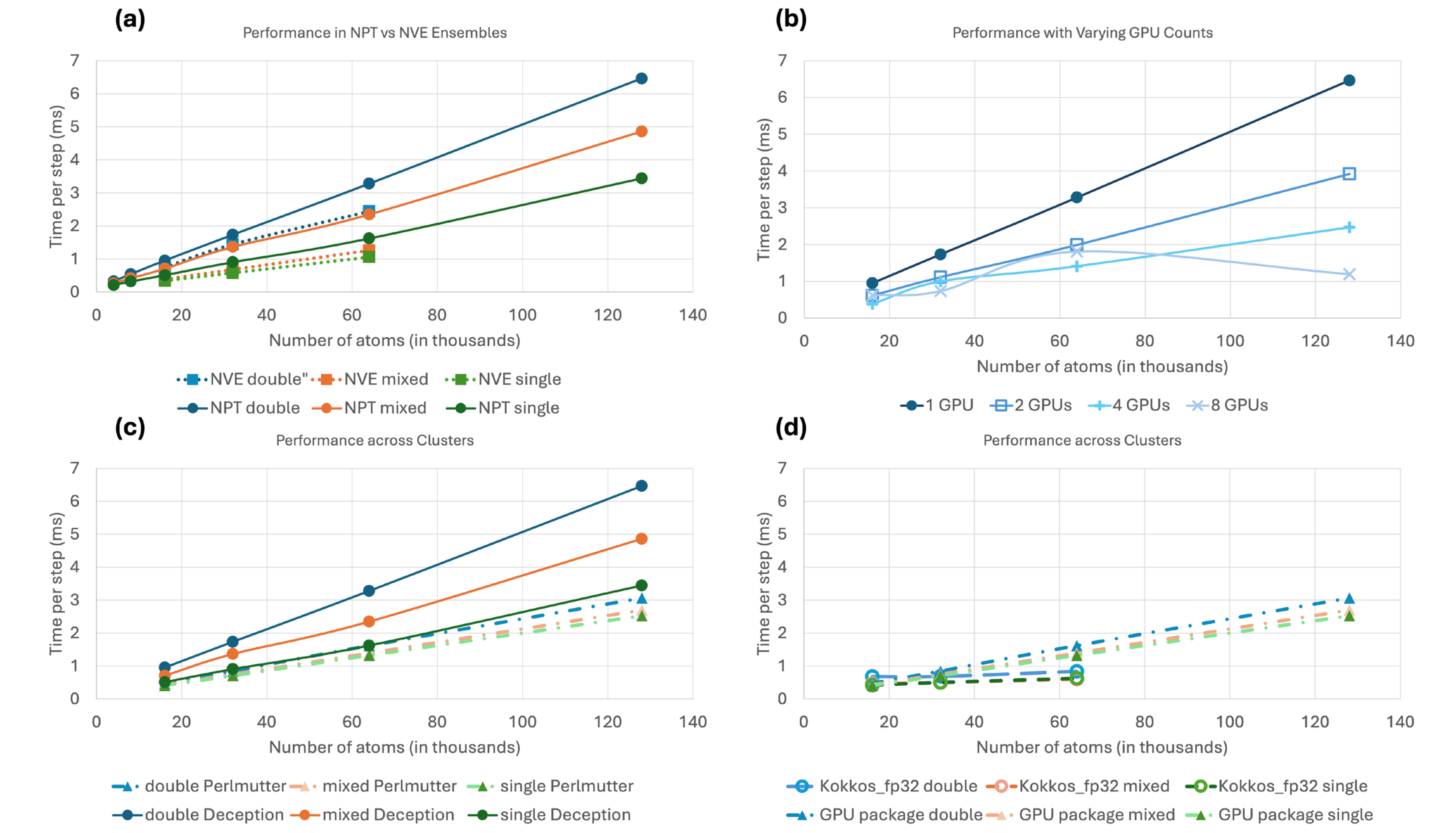

Figure 3. Performance comparisons: (a) NPT versus NVE; (b) different numbers of GPUs; (c) Perlmutter versus Deception; (d) GPU package versus kokkos_fp32. Results obtained with single, mixed, and double numerical precision formats are shown in each case.

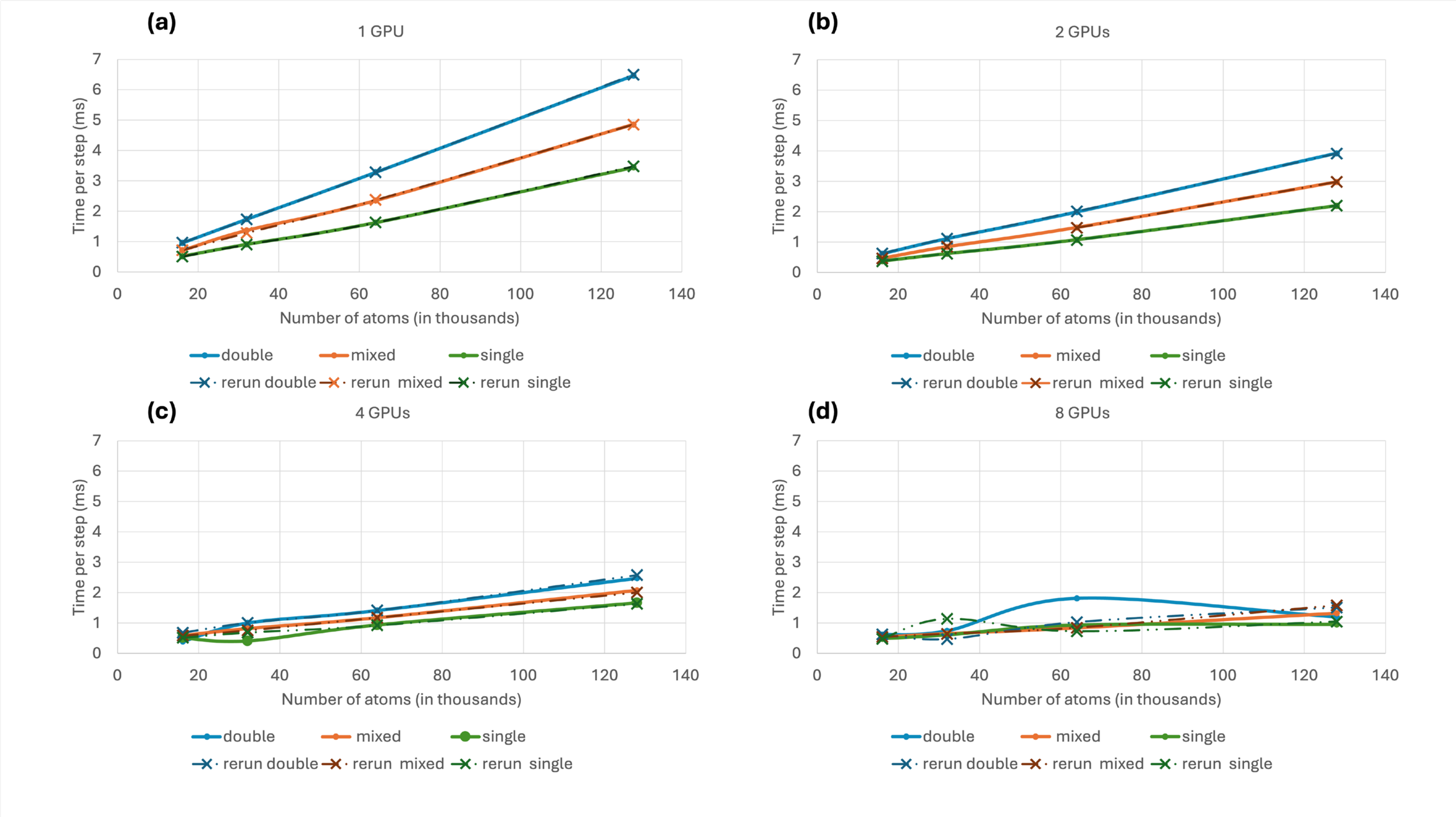

Figure 4 Reproducibility of simulations performed with 1, 2, 4, or 8 GPUs (panels a–d, respectively). Solid and dash-dot-dot lines indicate the initial and duplicate tests, respectively.

Performance was also evaluated on different machines. Figure 3(c) compares results from two clusters: Perlmutter (NERSC supercomputer with NVIDIA A100 GPUs) and Deception (PNNL supercomputer with RTX 2080 Ti GPUs). The results obtained with both clusters exhibited the same trend of increasing performance with reduced numerical precision with performance on Perlmutter being higher than on Deception.

We also compared the GPU package and Kokkos_fp32 using NPT simulations on a single GPU on Perlmutter. The GPU package is a stable and widely used GPU accelerator in LAMMPS for NVIDIA GPUs, supporting single, mixed, and double precision formats. It provides GPU implementations of many functional forms for pairwise interactions. It also accelerates parts of the k-space calculation needed to compute long-range electrostatic interactions. Kokkos is a C++ programming model for writing performance-portable applications on major HPC platforms, providing abstractions for parallel execution and data management. Kokkos_fp32 is an experimental extension of the Kokkos backend for LAMMPS aimed at enabling single- and mixed-precision simulations. It was suggested to us by NVIDIA developers and remains under active testing. For the same system size, Kokkos_fp32 was significantly faster than the GPU package (Figure 3(d)). Performance improved with reduced precision, although mixed and single precision formats resulted in almost identical timings with Kokkos_fp32.

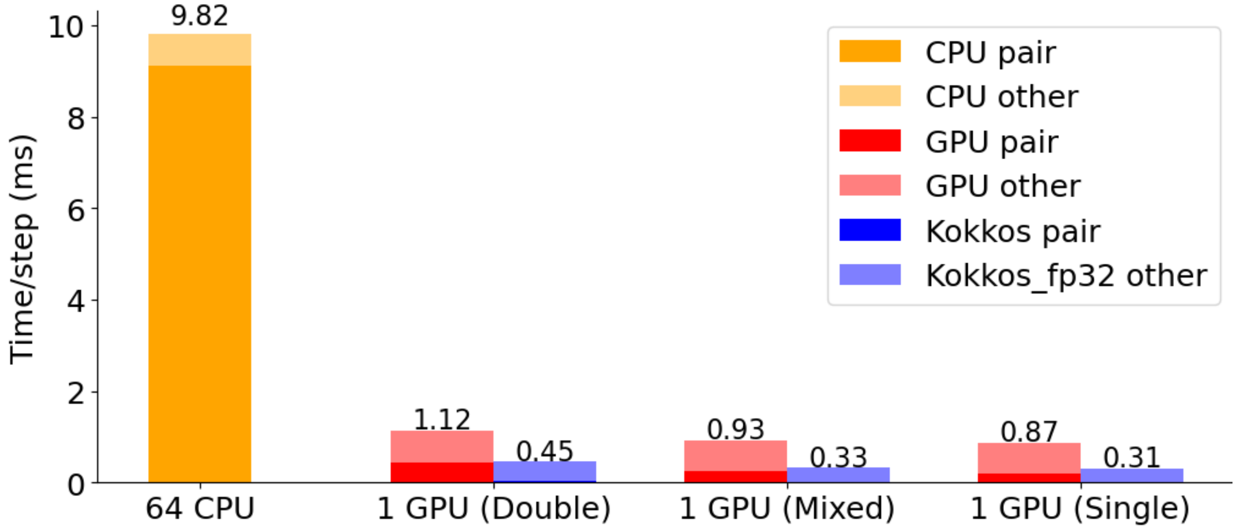

Figure 5. Liquid Ar system (64,000 atoms) at 100 K in the NVT ensemble. Effect of numerical precision on total time per step for LAMMPS GPU package and Kokkos_fp32. Timing obtained for the same system on 64 CPUs (double precision) is shown for comparison.

Figure 5 illustrates the main performance improvements achieved by using GPUs compared to CPUs, by using Kokkos_fp32 compared to the GPU package, and by lowering the numerical precision. To allow comparison with the results of the surrogate model developed in this work (see Section 3.0), these simulations were performed with a 64,000-atom system in the NVT ensemble at 100 K (i.e., for liquid Ar). To evaluate the impact of reduced precision on predicted properties of the liquid Ar system, we computed the Ar–Ar radial distribution functions (RDF) from MD trajectories generated using LAMMPS's CPU package, the GPU package (single precision), and Kokkos_fp32 (single precision). As shown in Figure 6, the RDFs completely overlapped, indicating that the use of lower numerical precision with either package did not affect accuracy.

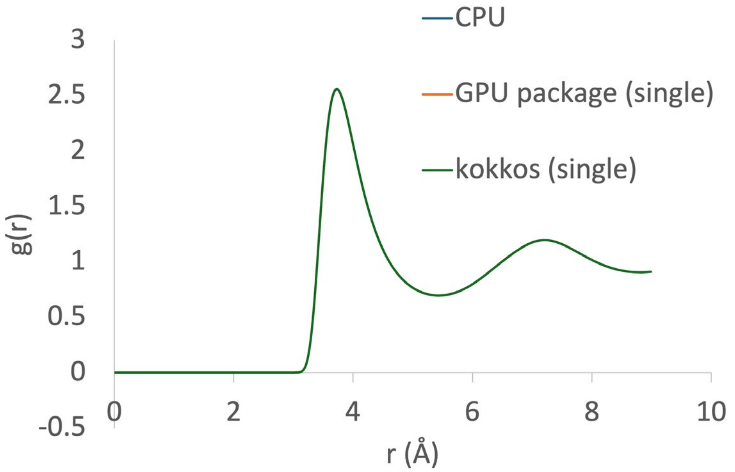

Figure 6. Radial distribution functions (RDFs) of argon obtained from MD simulations performed with LAMMPS (CPU), its GPU package (single precision), and Kokkos_fp32 (single precision).

2.2 NaCl System: k-Space Calculation

Following the tests performed with solid and liquid Ar and reported in the previous section, this section describes the software-level optimization performed with the NaCl system. While MD simulations of Ar only involve pairwise short-range interaction calculations because Ar atoms are neutral, simulating NaCl requires computing long-range electrostatic/Coulombic interactions between ions with formal charges of +1 or −1 (Eq. 1-1). Computing Coulombic interactions involves a reciprocal-space (k-space) calculation using fast Fourier transform (FFT). NaCl is therefore a more complex and computationally demanding system. All the simulations reported in this section were performed at 298 K using Kokkos_fp32 in the NPT ensemble at 298 K.

Two FFT libraries can be used with Kokkos_fp32: CUFFT, NVIDIA's GPU-accelerated fast Fourier transform library, and KISS FFT, a lightweight CPU-based library without GPU support. As shown in Figure 7, CUFFT achieves a large speedup compared to KISS FFT, while numerical precision has a small to negligible effect on FFT performance for a system with 64,000 atoms. The effects of the FFT library and numerical precision on the NaCl predicted structure were evaluated using Na–Cl RDFs (Figure 8). The RDFs obtained with MD trajectories computed with double-precision KISS and both double- and single-precision CUFFT overlap. This result indicates that the choice of FFT library and numerical precision do not affect the structural accuracy of the simulations.

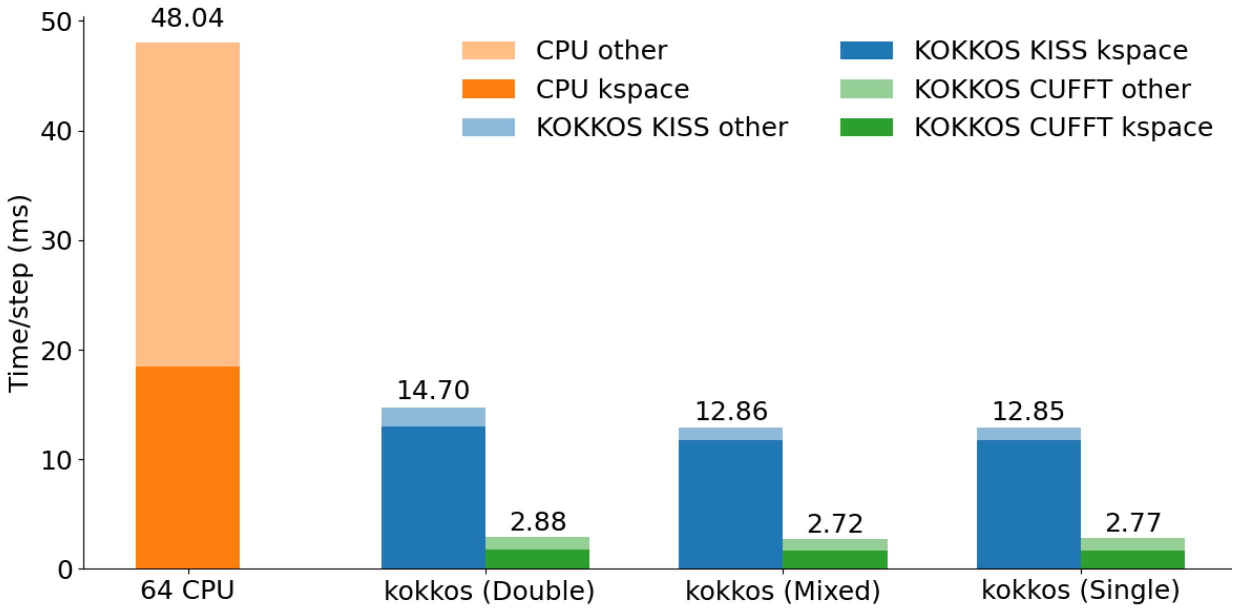

Figure 7. Performance for NaCl (64,000 atoms) using 64 CPUs and Kokkos_fp32 with KISS and CUFFT libraries in double, mixed, or single precision.

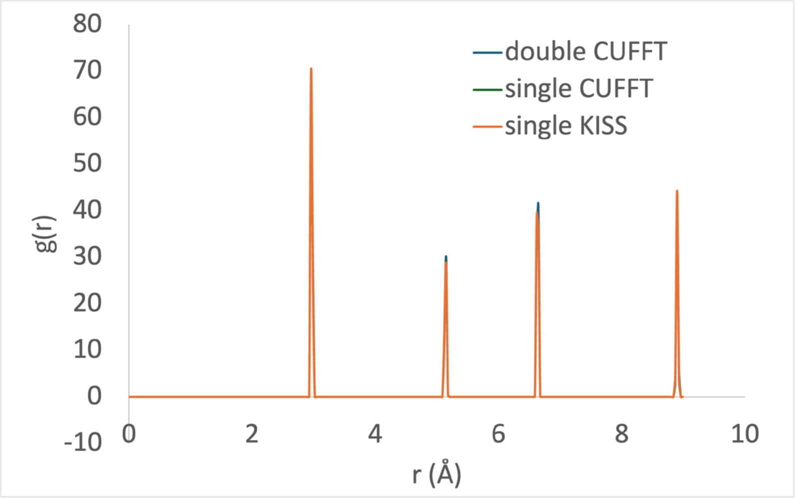

Figure 8. RDFs for Na and Cl computed with Kokkos_fp32 using CUFFT (double and single precision) and KISS (single precision).

Tests as a function of system size were performed using Kokkos_fp32 and the CUFFT library (Figure 9). For small systems (<64,000 atoms), different precisions exhibited similar performance for both the pairwise and k-space calculations. For larger systems, the time spent on pairwise interactions remained similar for the different precision formats, whereas differences in k-space performance became evident beyond 128,000 atoms. For the system with 256,000 atoms, the k-space calculation was 22.4% faster using single precision than using double precision. When the system size is not large enough to fully saturate the GPU, performance becomes limited by high latency and tail effects.

For the 256,000-atom system, the mixed-precision results deviated slightly from expectation. The pairwise interaction calculation took ~29% longer than the other two precisions, while double and single precision showed similar performance. It is difficult to determine whether this behavior is related to tail effects without further testing. According to the developers, there may be several superfluous and silent float-double conversions that slow down compute-bound kernels. The developers are actively working on addressing this issue.

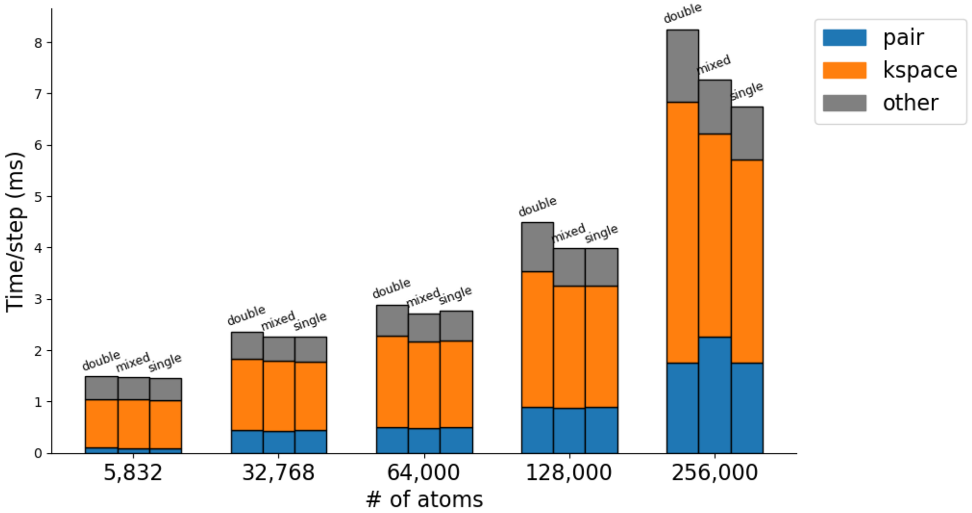

Figure 9. Performance with different numerical precision formats for systems of increasing size. Simulation time per step is decomposed into pair, k-space, and other contributions.

2.3 Challenges in Implementing Mixed-precision Optimizations

Impact of mixed-precision optimizations can be nuanced; performance issues stemming from suboptimal occupancy (i.e., inactive warps or groups of threads due to work assignment) and load imbalance (nontrivial number of idle threads waiting for the rest at any particular time) can diminish prospective gains from mixed-precision computations. These issues remain mostly hidden (despite major GPU acceleration in comparison to CPU), unless thorough profiling and configuration studies are conducted to quantify the trade-offs by increasing the system sizes and assigning an appropriate number of GPU threads per atom to enhance the occupancy (e.g., roughly a system with 27,000 atoms will be needed to fully occupy an NVIDIA A100 GPU). For smaller systems, several threads can be assigned on disparate symmetric multiprocessors of a single GPU, utilizing multiple processes per GPU via technologies such as CUDA MPS (multi-process service). Controlling the number of GPU threads per atom can improve the load imbalance (by balancing the tasks per thread) but would require coordination with the number of neighbors processed per thread; an imbalance between them can further exacerbate the load imbalance (due to vast disparities in neighbor computations between groups of threads).

After improving the GPU occupancy using the steps outlined above, one would be poised to extract further improvements owing to mixed-precision usage. Issues such as silent precision conversions by compilers/runtimes (allowed by C++ language standard) can impact the performance (we have observed such INT4 quantization limitations in JAX). Ultimately, MD software packages are designed for generality and portability; we need to specifically co-design newer mixed-precision kernels on current GPUs.

3.0 Software-level Optimization with Surrogate Modeling

We developed a data-driven surrogate for liquid argon forces that exploits locality and symmetry while remaining computationally lightweight. The force model is a compact pairwise multilayer perceptron (MLP) with widths [1, 64, 64, 64, 1] and tanh activations in the hidden layers. Its input is the normalized interatomic distance $s = \left|\left|r_{ij}\right|\right| / r_c$, and its output is a scalar weight applied along the unit displacement vector. For each atom, forces are obtained by summing $weight \times unit\left(r_{ij}\right)$ over neighbors within a cutoff $r_c = 8.5$ Å, with at most K = 128 neighbors per atom. Periodic boundary conditions enforce translational invariance; permutation invariance arises from the set-wise summation; and the directional aggregation makes the force rotate as a vector under rigid rotations. With a co-rotated periodic cell, this yields an SE(3)-equivariant force mapping under global rigid motions.

Training drew mini-batches consisting of one frame (10 ps intervals) and 200 randomly selected particles (Figure 10). Target forces were standardized using the training-set mean and standard deviation and were de-standardized at inference. The model was optimized with Adam (Kingma and Ba, 2014). For validation, we computed velocity Verlet rollouts using the same timestep used in the MD simulation ($dt = 10^{-3}\ ps$) and evaluated force fidelity using root mean square error (RMSE) and predicted forces ($\left|F_{pred}\right|$) versus reference forces ($|F_{true}|$) correlations (Figure 10). We also evaluated structural and energy fidelity using Ar–Ar RDF and potential energy ensemble average, respectively (Figure 11).

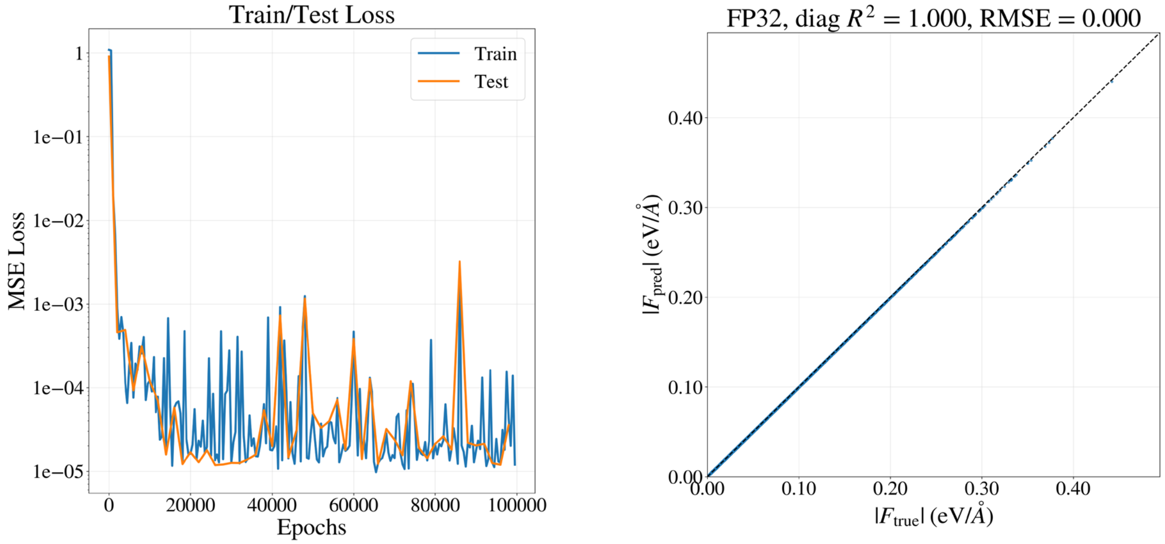

Figure 10. Model optimization and force fidelity. Left: Training and test mean-squared error (MSE) versus optimization steps. Right: FP32 force-magnitude correlation $\left|F_{pred}\right|$ versus $|F_{true}|$ on held-out frames with the diagonal y = x shown as reference; up to 50,000 samples are randomly subsampled for visibility, and the panel title reports the diagonal R^2 and RMSE. Axes are in eV/Å.

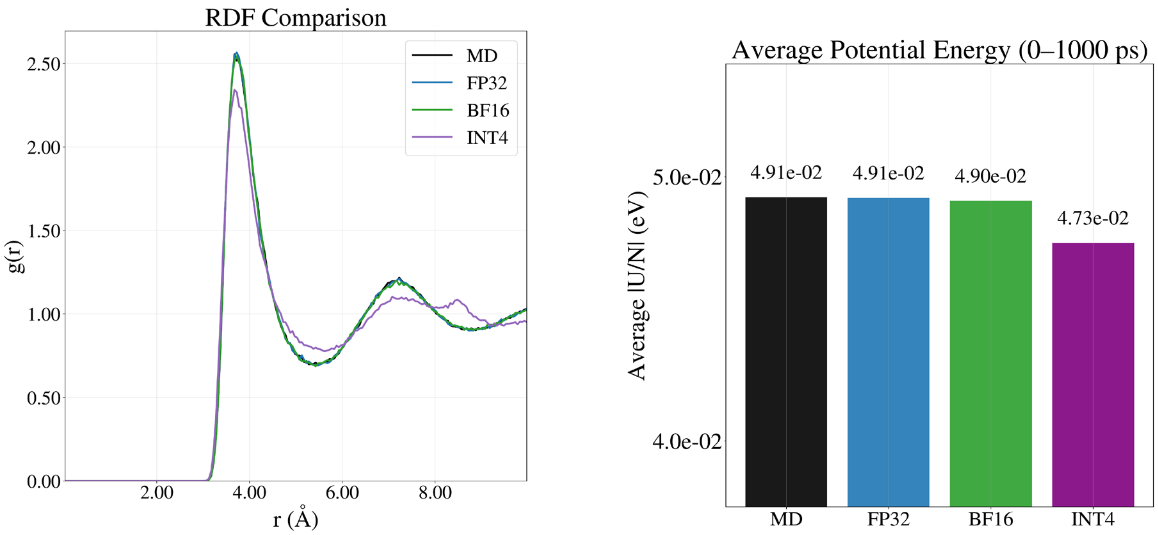

Figure 11. Structural and energy fidelity. Left: Ar–Ar RDFs computed from MD trajectories (“MD”), surrogate model (“FP32”), and surrogate model quantized using BF16 (“BF16”) or INT4 (“INT4”) numerical precision formats. Right: Potential energy ensemble averages.

4.0 Hardware Utilization Optimization with Post-training Quantization

Quantization experiments for the liquid-argon surrogate were run on the same hardware utilized in Section 3.0. We evaluated BF16 (floating-point) and INT4 (emulated, weight-only). Figure 12 shows force-magnitude correlations on held-out frames. The diagonal R^2 was 1.000 (FP32), 0.998 (BF16), and 0.944 (INT4), with corresponding RMSE of <0.001, 0.002 and 0.010, and 9.61×10^{-3} eV/Å, respectively. BF16 therefore preserved force fidelity close to FP32, whereas INT4 showed a clear loss consistent with the structural and energetic deviations. Figure 11 shows the structural and energetic comparisons: the RDF overlays matched the MD baseline, and the average potential energy over 0–1000 ps was unchanged within plotting resolution for BF16, whereas INT4 exhibited visible deviations. BF16 also reduced the force-evaluation compute time by ~25–27% relative to FP32 across system sizes equal to (2×, 4×, 8×, 10×) for a base system with 6,078 atoms. For BF16, weights are cast to bfloat16 at export and inference uses bfloat16 matrix multiplication precision; for portability, weights are stored as float32 on disk and cast at load time.

Figure 13 summarizes the runtime scaling: the compute sub-step benefits from BF16 while the neighbor-search component remains largely unchanged and dominates iteration time unless efficient neighbor lists are used. This neighbor-search-dominated profile is consistent with prior reports (Li et al., 2022).

Attempts to accelerate inference with INT4 on NVIDIA H100 (Hopper) did not succeed in our tests: a true 4-bit Tensor Core path was not engaged, exported INT4 weights were effectively quantize–dequantize at export, and inference fell back to floating-point matrix multiplications, so no speedup was observed. Enabling real INT4 acceleration will require int4×int4→int32 kernels exposed through JAX/XLA; we will revisit this when such support is available and after increasing compute intensity and neighbor reuse so that any gains are measurable. Even with working INT4 kernels, the present configuration's small GEMMs and dominant, precision-insensitive neighbor-search cost (~33–40 ms/iteration) would limit end-to-end improvement. In this report, INT4 is therefore used only to assess robustness of predictions, not performance.

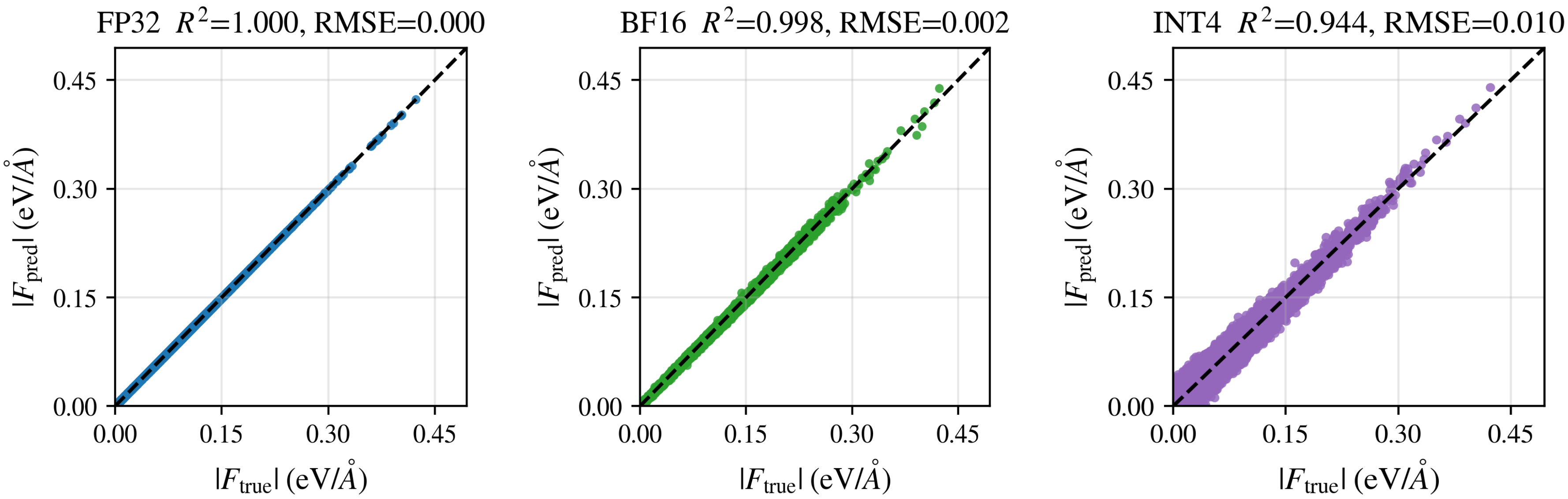

Figure 12. Force correlation for the tested quantized models. Left: FP32, Middle: BF16, Right: INT4; each panel shows $|F_{pred}|$ vs $|F_{true}|$ with diagonal reference and panel titles reporting diag R^2 and RMSE. The INT4 panel exhibits visibly larger scatter away from the diagonal and degraded metrics, compared with FP32 and BF16.

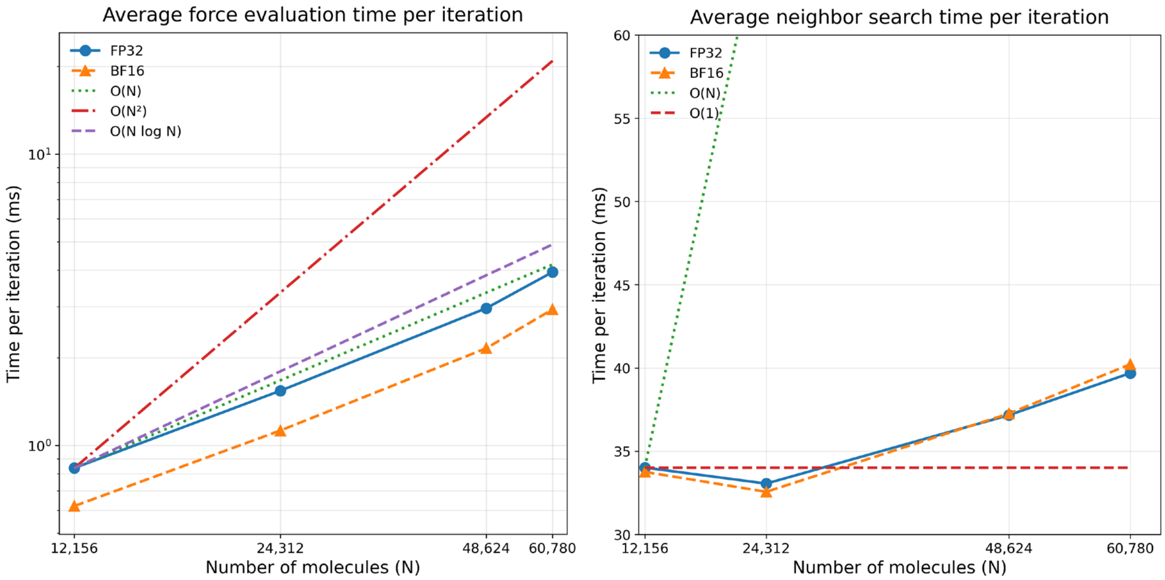

Figure 13. Runtime scaling with system size. Left: Average force-evaluation time per iteration (log–log axes) with FP32 and BF16, together with reference guides for O(N), O(N^2), and O($N \log N$) anchored at the smallest system. Right: Average neighbor-search time per iteration (linear axes) with FP32 and BF16, together with O(N) and O(1) guides. BF16 consistently accelerates the compute substep by roughly 25–27%, whereas neighbor search is largely insensitive to precision and dominates end-to-end iteration time unless efficient neighbor lists are used.

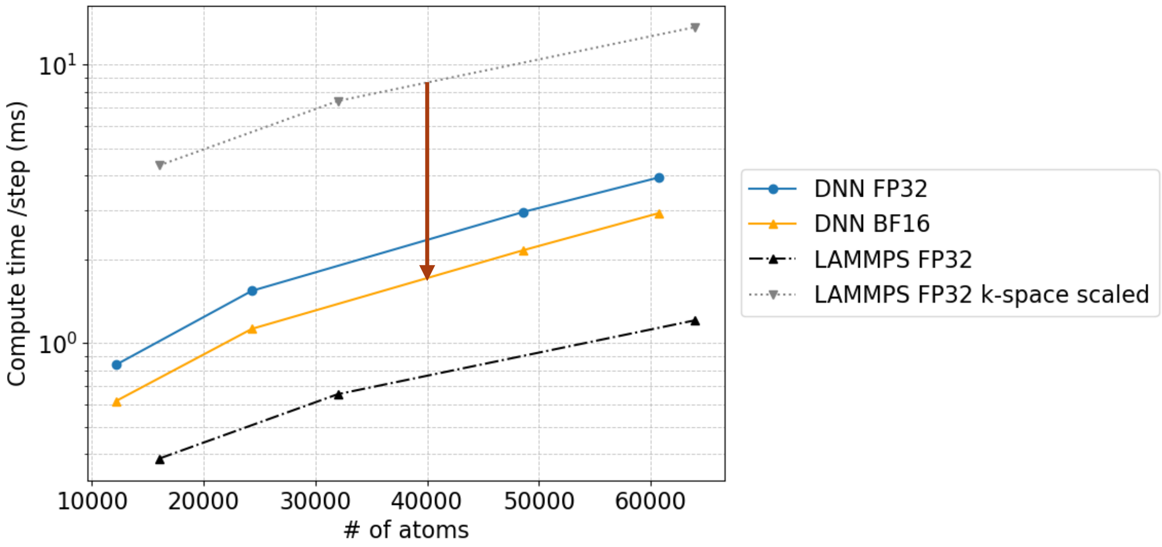

Figure 14. Comparison of force evaluation times and estimation of speedup achievable with DNN surrogate model of pairwise and electrostatic interactions (down arrow). Force evaluation times per step are shown for pairwise calculation with trained DNN (DNN FP32) and quantized DNN (DNN BF16) on NVIDIA H100 GPU together with times for pairwise calculation with LAMMPS-Kokkos_fp32 on NVIDIA A100 GPU (LAMMPS FP32) and estimated times for combined pairwise and k-space calculations with LAMMPS-Kokkos_fp32 on NVIDIA H100 GPU (LAMMPS FP32 k-space scaled) based on timings obtained on NVIDIA A100 GPU.

5.0 Recommendations for Future Work

Future work should focus on improving GPU utilization. Larger system sizes (≥500,000 atoms) should be tested to avoid latency and tail effects. Mixed-precision simulations of large systems should be examined in detail to assess numerical stability. For smaller systems, enabling neighbor thread parallelism may help overcome the size limitation. However, this approach can introduce load-balancing challenges and reduce cache reuse and therefore must be evaluated with caution. Additional strategies, such as running multiple simultaneous simulations using MPS or multi-instance GPU, should also be evaluated.

This work focused on Ar, a neutral atomic system requiring only pairwise interactions for force calculations, and NaCl, a crystalline system, for which force calculations also include long-range electrostatic interactions. However, forcefields for MD simulation can involve more complex interaction types, such as two- (bonds), three- (angles), and four- (torsions) body interactions. These complex interactions typically slow down MD simulations. In contrast, force calculation timings of DNN surrogate models should not be affected by the complexity of the forcefields they are replacing. Consequently, more complex forcefields are expected to translate to greater speedup afforded by the surrogate model. Therefore, future work should extend the DNN surrogate model developed in this work to encompass long-range electrostatic interactions as well as two-, three-, and four-body interactions, which would allow simulation of molecular systems. Future work could also apply this approach to ab initio MD (AIMD) simulation where significant efficiency gains are expected through the generation of surrogate models that retain the chemical accuracy of AIMD simulations. Finally, DNN surrogate models should be extended to graph neural networks (GNNs) to better capture information transfer between atoms.

6.0 References

Behler, J., 2021. Four generations of high-dimensional neural network potentials. Chemical Reviews, 121: 10037-10072. 10.1021/acs.chemrev.0c00868:10.1021/acs.chemrev.0c00868

Jia, W., Wang, H., Chen, M., Lu, D., Lin, L., Car, R., E, W., Zhang, L., 2020. Pushing the limit of molecular dynamics with ab initio accuracy to 100 million atoms with machine learning. In: Conference, I. (Editor), SC20: International conference for high performance computing, networking, storage and analysis.

Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. arXiv. 10.48550/arXiv.1412.6980:10.48550/arXiv.1412.6980

Le Grand, S., Gotz, A.W., Walker, R.C., 2013. SPFP: Speed without compromise - A mixed precision model for GPU accelerated molecular dynamics simulations. Computer Physics Communications, 184: 374-380. 10.1016/j.cpc.2012.09.022:10.1016/j.cpc.2012.09.022

Li, Z., Meidani, K., Yadav, P., Farimani, A.B., 2022. Graph neural networks accelerated molecular dynamics. Journal of Chemical Physics, 156: 144103. 10.1063/5.0083060:10.1063/5.0083060

Plante, A., Shore, D.M., Morra, G., Khelashvili, G., Weinstein, H., 2019. A machine learning approach for the discovery of ligand-specific functional mechanisms of GPCRs. Molecules, 24: 2097. 10.3390/molecules24112097:10.3390/molecules24112097

Trizio, E., Parrinello, M., 2021. From enhanced sampling to reaction profiles. Journal of Physical Chemistry Letters, 12: 8621-8626. 10.1021/acs.jpclett.1c02317:10.1021/acs.jpclett.1c02317

Appendix A – Mixed Numerical Precision Formats

Lower-precision numerical formats use less memory (since memory and bandwidth are scarce in computing, reducing memory usage for compute-intensive workloads almost always improves the overall bandwidth) – e.g., see 16-bit floating point (FP16) vs. 32-bit floating point (FP32) (Figure A- 1), in contrast, several traditional science applications such as MD simulations often use 64-bit floating point, and lack broader support for lower precision, even when deployed on mixed-precision capable hardware such as modern NVIDIA GPUs. The ultimate goal is to trade-off accuracy for performance (i.e., owing to enhanced bandwidth and efficient mixed-precision enabled logic units in contemporary GPUs). In the Machine Learning world, higher precision is almost always unnecessary, and existing methods routinely utilize computations with half-precision and below.

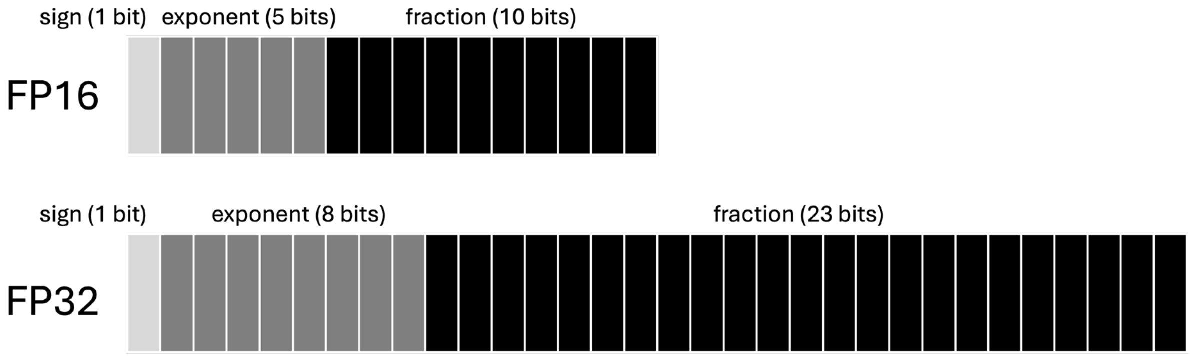

Figure A- 1: FP16 (half-precision) vs. FP 32 (single-precision).

In terms of representing decimal numbers, FP32 low range is 10^{-38} whereas for FP16 it is 10^{-8} (smallest number) – therefore, FP16 must trade-off accuracy with enhanced computation performance. But, often, both performance and numerical stability is required. As such, several numerical formats have been proposed recently to bridge the precision gap between FP16 and FP32. For e.g., brain-floating format or BF16 also uses 16 bits like FP16 but uses 8 bits (+3 vs. FP16) for exponent and 7 bits (−3 vs. FP16) for fraction. BF16 low range is 10^{-38} meaning that its performance is like that of FP16 and numerical stability matches FP32.

Pacific Northwest National Laboratory

902 Battelle Boulevard
P.O. Box 999
Richland, WA 99354

1-888-375-PNNL (7665)

www.pnnl.gov