Freie Universität Berlin

A fast implementation of deep neural-network potentials for molecular dynamics simulations of alloys

by

Sandip Kumar Sah

Master Thesis in Computational Science

Submission: October 26, 2025 Supervisor: Prof. Dr. Felix Höfling

Freie Universität Berlin | Department of Mathematics and Computer Science
Institute of Mathematics

Statutory Declaration

Family Name, Given/First Name	Sah, Sandip Kumar
Matriculation number	5589263
Kind of thesis submitted	Master Thesis

English: Declaration of Authorship

I hereby declare that the thesis submitted was created and written solely by myself without any external support. Any sources, direct or indirect, are marked as such. I am aware of the fact that the contents of the thesis in digital form may be revised with regard to usage of unauthorized aid as well as whether the whole or parts of it may be identified as plagiarism. I do agree my work to be entered into a database for it to be compared with existing sources, where it will remain in order to enable further comparisons with future theses. This does not grant any rights of reproduction and usage, however.

This document was neither presented to any other examination board nor has it been published.

German: Erklärung der Autorenschaft (Urheberschaft)

Ich erkläre hiermit, dass die vorliegende Arbeit ohne fremde Hilfe ausschließlich von mir erstellt und geschrieben worden ist. Jedwede verwendeten Quellen, direkter oder indirekter Art, sind als solche kenntlich gemacht worden. Mir ist die Tatsache bewusst, dass der Inhalt der Thesis in digitaler Form geprüft werden kann im Hinblick darauf, ob es sich ganz oder in Teilen um ein Plagiat handelt. Ich bin damit einverstanden, dass meine Arbeit in einer Datenbank eingegeben werden kann, um mit bereits bestehenden Quellen verglichen zu werden und dort auch verbleibt, um mit zukünftigen Arbeiten verglichen werden zu können. Dies berechtigt jedoch nicht zur Verwendung oder Vervielfältigung.

Diese Arbeit wurde noch keiner anderen Prüfungsbehörde vorgelegt noch wurde sie bisher veröffentlicht.

.....

Date, Signature

Abstract

A concise 200–300 word summary of:

- Motivation (accelerating atomistic simulations using neural network potentials)
- Goal (integrating DeepMD potential calculation into HALMD)
- Method (extracting weights, replicating inference in C++/CUDA)
- Key results and performance (accuracy vs. speed trade-off)
- Conclusions

Contents

1 Introduction

1.1 Motivation

1.2 Objectives and Scope

1.3 Structure of the Thesis

This thesis is organized into six main sections, each addressing a distinct aspect of the integration of deep neural-network potentials into the HALMD simulation framework for binary alloy systems.

Section 1: Introduction presents the motivation for employing machine-learned interatomic potentials in molecular dynamics, highlighting the limitations of traditional analytical models and the potential of deep learning to achieve *ab initio*-level accuracy at a fraction of the computational cost. The objectives and scope of the work are defined, followed by an outline of the thesis structure.

Section 2: Background introduces the theoretical and computational foundations of the work. It provides an overview of the HALMD software, describing its modular design, GPU acceleration, and classical molecular dynamics formulation. The section also discusses the Deep Potential Molecular Dynamics (DeepMD) framework, including the principles of neural-network potentials, descriptor construction, and the overall architecture of the Deep Potential model for multi-species systems such as alloys.

Section 3: Methodology details the integration process of the DeepMD potential into HALMD. It explains the extraction of model parameters (weights, biases, and descriptors) from a trained DeepMD model, the reconstruction of the descriptor pipeline (R and G matrices), and the implementation of the neural network inference and force computation within HALMD. Special attention is given to the treatment of multiple atomic species and the efficient GPU-based execution of the potential evaluation.

Section 4: Results presents the outcomes of the integration. This includes validation of the energy and force predictions against reference DeepMD-kit calculations, performance benchmarks of the HALMD implementation on GPU hardware, and case studies of binary alloy systems. The accuracy, scalability, and computational efficiency of the implementation are systematically analyzed.

Section 5: Discussion interprets the results, addressing the accuracy–performance trade-offs, sources of deviation between HALMD and DeepMD outputs, and the numerical stability of the approach. It also highlights the implications of using neural-network potentials in multi-component systems.

Section 6: Conclusions and Future Work summarizes the key contributions of the thesis and discusses possible directions for future development. Potential extensions include optimized GPU kernels, support for additional descriptor types, and extension of the implementation to more complex alloy systems beyond binaries.

Overall, the structure of the thesis is designed to guide the reader from the motivation and theoretical foundations through the detailed implementation and validation of a high-performance, machine-learned potential within the HALMD framework.

2 Background

2.1 HALMD Software

2.2 Neural Network Potentials

2.3 Deep Potential Molecular Dynamics (DeepMD)

3 Methodology

3.1 Overview of Implementation

The integration of the Deep Potential Molecular Dynamics (DeepMD) model into the High-Accuracy Large-scale Molecular Dynamics (HALMD) framework constitutes the core technical contribution of this thesis. The goal of the implementation is to enable HALMD to evaluate interatomic forces and potential energies directly from a trained DeepMD-kit v2 model—without relying on TensorFlow or the original Python runtime—thus allowing fully GPU-accelerated molecular dynamics simulations of binary alloy systems within HALMD's native C++ and CUDA environment.

The implementation process follows a modular workflow that bridges the DeepMD and HALMD architectures. It consists of two main components: a Python-based parameter extraction stage and a C++/CUDA-based inference stage.

1. Parameter Extraction (Python side). The first step involves extracting the neural network parameters—weights, biases, and embedding coefficients—from the trained DeepMD-kit v2 model stored in the `frozen_model.pb` file. This file contains a serialized TensorFlow computation graph representing the trained descriptor construction, filter (embedding) network, and fitting network. A custom Python utility script was developed using the TensorFlow API to traverse the graph structure, locate all relevant trainable variables, and export them in a structured and human-readable format (JSON or NumPy arrays). The extraction process ensures that all model parameters are preserved with full numerical precision, including the layer dimensions, activation functions, and cutoff parameters defined in the associated `input.json` file.

2. Inference and Integration (C++/CUDA side). Once extracted, the parameters are imported into HALMD's modular potential interface. A new potential class, referred to as `deepmd_potential`, was implemented in C++ to replicate the DeepMD inference process. This class reconstructs the descriptor pipeline—computation of R- and G-matrices, application of smooth cutoff functions, and formation of atomic descriptor vectors $\mathbf{D}_i$—directly within HALMD. The descriptor output is then propagated through the fully connected neural networks using the extracted weights and biases to compute per-atom energies. Forces are obtained analytically by differentiating the energy with respect to atomic coordinates using explicit derivative propagation through both the descriptor and network layers.

HALMD's existing GPU infrastructure was leveraged to parallelize the most computationally demanding operations, including neighbor-list construction, descriptor evaluation, and dense matrix multiplications within the neural network layers. By combining DeepMD's learned potential surface with HALMD's optimized CUDA kernels, the resulting implementation achieves efficient large-scale simulations of binary alloy systems while maintaining consistency with the original DeepMD-kit v2 predictions.

3. Verification and Validation. To ensure correctness, the C++ implementation was systematically validated against the original DeepMD Python inference. Energy and force predictions were compared for identical test configurations of binary alloy systems, and discrepancies were measured to confirm numerical consistency. Once verified, the integrated DeepMD potential in HALMD was used for production-level simulations, enabling the exploration of alloy thermodynamics and dynamics at near *ab initio* accuracy.

Overall, the implementation strategy is designed to separate data extraction and inference logic. The Python stage ensures model transparency and compatibility with DeepMD-kit v2, while the C++ stage enables real-time evaluation within HALMD's GPU-accelerated molecular dynamics engine. This modular design provides a scalable foundation for future extensions, including support for additional descriptor types or multi-GPU parallelization.

3.2 Model Parameter Extraction

3.2.1 Frozen Model Structure

3.2.2 Extraction Procedure

3.3 Coordinate System Extension

Representation of atomic environments in HALMD vs. DeepMD. Neighbor list construction and cutoff scheme. Transformation of Cartesian coordinates to local reference frames.

3.4 Computation of R and G Matrices

3.4.1 R Matrix

3.4.2 G Matrix

3.5 Descriptor Computation

3.6 Potential Energy Calculation

3.7 Force Computation

3.7.1 Descriptor Derivative

3.7.2 Network Derivative

4 Results

4.1 Verification

4.2 Performance Evaluation

4.3 Case Study: Binary Alloy System

5 Discussion

6 Conclusions and Future Work