

NNMD: ПРОГРАММНЫЙ ПАКЕТ НА PYTHON ДЛЯ ОБУЧЕНИЯ МЕЖАТОМНЫХ ПОТЕНЦИАЛОВ С ИСПОЛЬЗОВАНИЕМ ФУНКЦИЙ БЕЛЕРА–ПАРРИНЕЛЛО

А. С. Будников^{1,*}, А. В. Романов^{1,**}

¹ Воронежский государственный университет, Воронеж, Россия

С развитием инструментов машинного обучения появилась возможность их применения в различных областях науки. Например, создав и обучив специально сконструированную нейронную сеть, можно получить значительное ускорение процесса вычислений по сравнению с *ab initio* молекулярной динамикой. Представлен программный пакет для создания, обучения и применения нейронных сетей Белера–Парринелло на языке программирования Python и фреймворке машинного обучения PyTorch.

Development of machine learning tools makes it possible to apply them in various fields of science. For example, by creating and training specially designed neural network it is possible to obtain a significant acceleration of computational process compared to *ab initio* molecular dynamics. This paper presents a Python programming package for creating, training, and applying Behler–Parrinello neural networks using PyTorch framework.

PACS: 89.20.Ff; 07.05.Mh

ВВЕДЕНИЕ

Развитие инструментов машинного обучения (Machine Learning — ML) за последние несколько лет привело к интеграции различных ML-алгоритмов, в том числе и нейронных сетей (Neural Network — NN), с гетерогенными вычислительными системами, а также к их внедрению в различные отрасли науки, в частности, вычислительную химию. Одним из примеров применения нейронных сетей в химии является моделирование химических процессов и новых материалов методом молекулярной динамики с машинно-обучаемыми потенциалами. Построив и обучив специально сконструированную модель нейронной сети, например, нейронную сеть Белера–Парринелло (Behler–Parrinello Neural Network — BPNN), можно переложить задачу подбора формы межатомных потенциалов, необходимой для молекулярно-динамического моделирования,

* E-mail: budnikov.andrey777@gmail.com

** E-mail: romanov_av@cs.vsu.ru

на вычислительную машину. Такой подход позволяет получать результат за адекватное время с возможностью повторного использования модели на других задачах и ее дообучения на новых данных. На сегодня представлены различные программные пакеты, реализующие данный метод, к примеру, DeePMD-kit [1], Fortnet [2], PiNN [3], SIMPLE-NN [4] и т. д.

Описывается программный пакет NNMD (Neural Networks for Molecular Dynamics) [5] pre-alpha версии 0.2.0. Пакет разрабатывается на языке Python для упрощения сопровождения и расширения пакета новыми наборами дескрипторов и моделями нейронных сетей. NNMD планируется использовать для ускорения исследований в области водородной энергетики, в частности, для изучения взаимодействия водорода с металлическими сплавами. В нем планируются инструменты для автоматической генерации датасетов, а также интерфейсы к популярным пакетам молекулярной динамики, например, ASE, Lammmps и Gromacs.

МАШИННО-ОБУЧАЕМЫЕ ПОТЕНЦИАЛЫ

Машинно-обучаемые потенциалы — это регрессионная модель, которая описывает зависимость между атомными конфигурациями системы и их полными энергиями [4]. Согласно классификации, предложенной Й. Белером [6], всего существует четыре поколения машинно-обучаемых потенциалов. Первое поколение таких моделей основано на архитектуре многослойного перцептрона с фиксированным числом входных нейронов, равной размерности атомной системы. Модели такого типа имеют свои ограничения: во-первых, потенциалы первого поколения могут предсказывать энергию только для систем низкой размерности, что не позволяет масштабировать модель на большие системы с высокой размерностью и разными типами атомов; во-вторых, при таком типе дескрипторов не достигается трансляционная и ротационная инвариантность получаемой энергии.

Второе поколение оказалось более применимым для моделирования больших систем. Оно отличается следующим: было сделано предположение, что большое число межатомных взаимодействий может быть описано через локальные атомные окружения, что позволит обучать ансамбли нейронных сетей, или нейронные сети высокой размерности, а также был предложен новый тип дескрипторов — атомно-центрированные симметричные функции Белера–Парринелло. Чтобы при вычислении функций и последующем обучении исключались малозначимые межатомные связи, в формулы вводится специальная функция отсечки:

$$f_c = \begin{cases} \frac{1}{2} \left(\cos \left(\frac{\pi R_{ij}}{R_c} \right) + 1 \right), & \text{если } R_{ij} \leq R_c, \\ 0, & \text{если } R_{ij} > R_c, \end{cases}$$

где R_c — радиус отсечки.

Использование потенциалов второго поколения позволяет помимо полной энергии достаточно точно определять силы на атомах и напряженность. Однако существенной проблемой является определение набора симметричных функций, который однозначно описывает локальное окружение атомов в рассматриваемой системе. Также использование функции отсечки приводит к локальности получаемых потенциалов, что не позволяет применять модель к системам с электростатическими взаимодействиями.

Модели потенциалов третьего и четвертого поколений в большинстве своем основаны на разного рода комбинациях двух или более нейронных сетей высокой размерности второго поколения и их модификаций, что решает проблему локальности, чтобы машинно-обучаемые потенциалы могли применяться для большего числа практических задач.

РЕАЛИЗАЦИЯ ML-ПОТЕНЦИАЛОВ ВТОРОГО ПОКОЛЕНИЯ

NNMD разрабатывается как проект с открытым исходным кодом, реализованный полностью на языке программирования Python с использованием фреймворка PyTorch для машинного обучения на GPU.

Пакет разбит на несколько модулей: `io` (input/output), `nn` (нейронные сети и связанные с ними инструменты), `features` (вычислительная часть пакета, дескрипторы), `md` (имплементация калькулятора для ASE). Благодаря модульной структуре пользователь может использовать только необходимые инструменты без необходимости взаимодействовать со всеми компонентами пакета, а также расширять и адаптировать код под конкретные нужды и задачи.

Процесс обучения с использованием пакета NNMD выглядит следующим образом (рис. 1): в начале процедуры считываются данные из файла формата ASE, GPAW или NumPy-архива (формат `.npz/.npy`). После этого запускается само обучение модели: получение характеристик атомной системы, вычисление функции потерь, обратное распространение ошибки и т. д. В результате создаются файлы с сохраненными весами подсетей в формате `.pt` и `log`-файлы, хранящие информацию о параметрах и результатах обучения.

Рассмотрим пример использования пакета NNMD на кристаллической структуре из 27 атомов лития. Датасет содержит 23 714 состояний системы, полученных в ходе отжига квантовой молекулярной динамики. Данные были получены на вычислительном кластере факультета компьютерных наук ВГУ, результаты получены на видеокарте NVIDIA GeForce RTX 3070.

Обычной практикой для подобных пакетов является выбор 50 функций (радиальных и угловых) для каждого атома системы. Для данной системы с учетом ограничений в виде объема доступной памяти на видеокарте было выбрано 12 радиальных и 2 угловые функции. Предва-

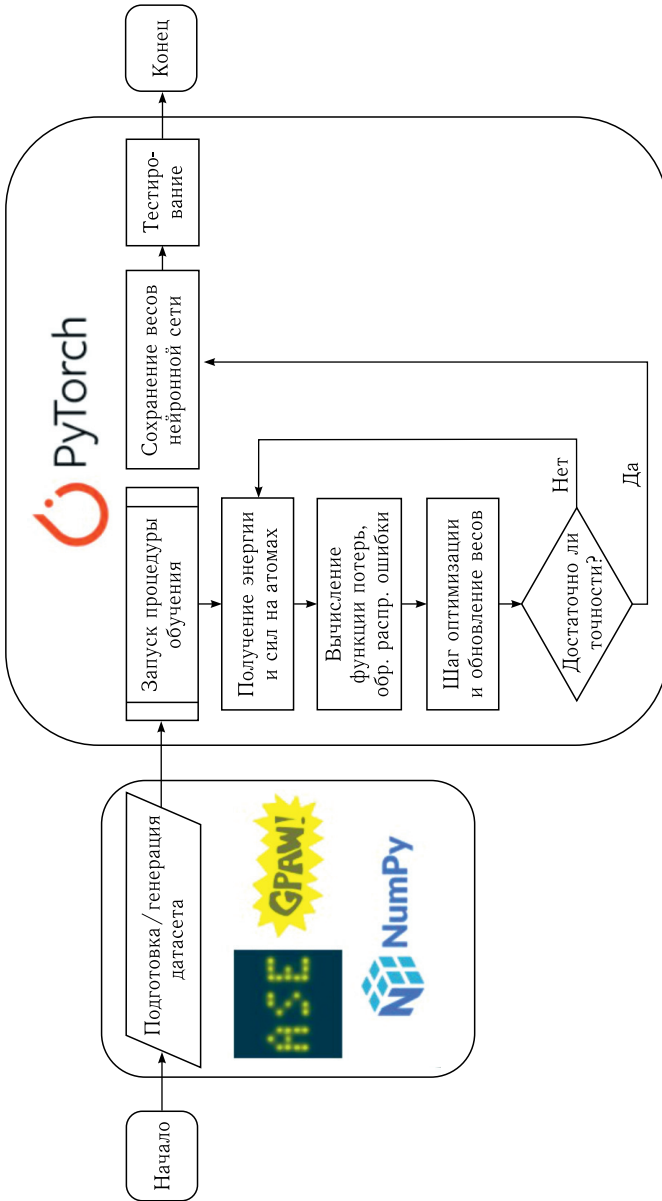


Рис. 1. Схема процедуры обучения в NNMD



Рис. 2. Сравнение предсказанной и фактической энергий

рительное моделирование показывает правильность выбранного направления: как видно из представленного на рис. 2 графика нормализованных предсказанной и полученной энергий на небольшом временном участке (например, с 15 860 по 16 863 фрейм) для наглядности, обученная модель воспроизводит динамику системы, среднеквадратичная ошибка по результатам составила около 0,001 эВ по энергии и 0,0003 эВ/Å по силам.

ЗАКЛЮЧЕНИЕ

Описан метод построения машинно-обучаемых потенциалов для молекулярно-динамического эксперимента, а также представлено актуальное состояние пакета NNMD.

В будущем планируется совершенствование реализации моделей машинно-обучаемых потенциалов второго поколения, представленной в пакете, и переход к моделям третьего поколения для работы с электростатическими взаимодействиями. Также программная часть будет изменена: будет реализована большая вариативность при обучении путем добавления большего числа настраиваемых аспектов обучения, например, расширение набора доступных оптимизаторов, в частности, добавление оптимизатора L-BFGS и планировщиков обучения, генерация параметров симметричных функций для автоматизации.

СПИСОК ЛИТЕРАТУРЫ

1. *Zeng J. et al.* DeePMD-kit v2: A Software Package for Deep Potential Models // J. Chem. Phys. 2023. V. 159. 054801.
2. *Van der Heide T. et al.* Fortnet, a Software Package for Training Behler–Parrinello Neural Networks // Comput. Phys. Commun. 2023. V. 284. 108580.
3. *Shao Y. et al.* PiNN: A Python Library for Building Atomic Neural Networks of Molecules and Materials // J. Chem. Inform. Model. 2020. V. 60, No. 3.
4. *Kyuhyun L., Dongsun Y., Wonseok J., Seungwu H.* SIMPLE-NN: An Efficient Package for Training and Executing Neural-Network Interatomic Potentials // Comput. Phys. Commun. 2019. V. 242. P. 95–103.
5. Проект NNMD. <https://github.com/ded-otshelnik/nnmd> (дата обращения 16.09.2025).
6. *Behler J.* Four Generations of High-Dimensional Neural Network Potentials // Chem. Rev. 2021. V. 121, No. 16. P. 10037–10072.