========================README.TXT=========================

**Modeling Enzyme Reaction and Mutation by Direct Machine Learning/Molecular Mechanics Simulations**

**Description of Data Files**

The main branch of Qeq-REANN contains three parts: code, mdData and model.

**code:**

It contains the training code of Qeq-REANN package and interface for inference model. Details are provided in the sections below.

**mdData**

It contains the reaction coordinate, energy in the ME scheme (Elow) and EE scheme (Ehigh) for harvested configurations. The analysis of free energy in manuscript is based on these data.

**model**

It contains the recommended parameters and configurations used for training the model. The final models for simulations are also provided.

**How to Use Qeq-REANN Package**

This is a modified version of the REANN package under the agreement of B. Jiang’s Group, encoded with the charge equilibrium (Qeq) method, with an additional functionality for training atomistic charges. The original copy of REANN package is available at the website <https://github.com/zhangylch/REANN/>.

The manual of original REANN package is also included in the path “code/Qeq-REANN\_core/manual”. Besides the original functionality of REANN, users may realize the training of point charges by following these steps:

1. **Prepare the environment**

The REANN Package is built based on PyTorch and uses the "opt\_einsum" package for optimizing einsum-like expressions frequently used in the calculation of the embedded density. In order to run the REANN package, users need to install PyTorch (version: 2.0.0) and its dependency environment based on the instructions on the PyTorch official website (https://pytorch.org/get-started/locally/) and the package named opt\_einsum (<https://optimized-einsum.readthedocs.io/en/stable/>). However, in our test, the Qeq-REANN identified a bug when using double-float type to train models in the version 2.1.0 of PyTorch.

1. **Prepare data**

In our scheme, surfaces of potential energy and point charges need to be trained, and there are two directories that users need to prepare respectively, in namely “train” and “test”, each of which includes a file “configuration”, used to preserve the required information including lattice parameters, periodic boundary conditions, configurations, energy/charge and 3 atomic forces (if needed). We provided the examples for PES and surface of point charges in the path “model/pes” and “model/chs” respectively.

1. **Construct a model**

Two sets of reference training parameters are also provided in the path “model/pes” and “model/chs” respectively. Specifically, start\_table = 5 for charges and start\_table = 6 for Qeq charges. Other parameters can be referred in the manual of REANN package.

After training, user can get the .pt file which is the inference model of Qeq-REANN.

**How to Build an Interface for the Amber Program**

Because Amber is commercial software, the modification of its codes in our scheme are not available unless reasonable request. Here, we only offer its external interface with Qeq-REANN. To build the interface for the Amber program, user needs to build a dynamic-link library to link the JIT inference model of Pytorch trained by Qeq-REANN to the main program sander in Amber, which is written in Fortran. The interface spans four programming languages: **PyTorch, C++, C,** and **Fortran**, which is realized via **Cmake** program. The directory of reann-testch\_omp is a trial version of OpenMP parallelism.

The original copy for inferring energy is available at the website <https://github.com/junfanxia/proj-reann-cpp2fortran-fixedcell>.

The compilation is relatively complicated and user needs to follow these steps strictly:

1. **Prepare the environment**

Here are the software requirements:

CMake 3.19.3

libtorch-CPU/GPU 1.12.1

gcc/g++/gfortran 8.5.0

CUDA(only for GPU) 11.3

OpenMP(optional, only for CPU) 4.5

1. **Compiling and Linking**

User needs to change the path of compilers in “build/build.sh” and the path of libtorch in “src/CmakeLists.txt” and “src/interfaces/CmakeLists.txt”. Then, execute Linux commands in the root directory of the interface:

cd build; sh build.sh; make

Then we can get the directories named “libs” and “modules” in the path of “build/”, which contains the dynamic-link library and header file in Fortran. User needs to replace original function call of ab initio calculations in the programs of molecular dynamics by the template in src/test.f90.

With these files, user can link them to Amber or other programs of molecular dynamics through adding the codes listed below in the Makefile like:

gfortran -fopenmp sander.o -I /home/shaxh/interface/reann-testen/build/modules/ -I /home/shaxh/interface/reann-testchParallel/build/modules/ -L /home/shaxh/interface/reann-testen/build/lib/ -L/home/shaxh/interface/reann-testchParallel/build/lib/ -I /home/shaxh/interface/reann-testen/build/modules/ -I /home/shaxh/interface/reann-testchParallel/build/modules/

Then the interface is successfully linked to Amber.

1. **Loading the Model**

User needs to prepare a “input\_reann” file to compensate for the missing parameters of the inference model (in the formation of .pt). It contains the information of cell, pbc, number of atoms, their species, maxtype and type species, which is provided in the path “code/Qeq-REANN Fortran interface” as well. With the “input\_reann” and .pt file, after the linking process, user can run the simulations with surfaces of Qeq-REANN.

**CONTACT INFORMATION**

Yanzi Zhou

School of Chemistry and Chemical Engineering, Nanjing University,

Nanjing, Jiangsu, 210023, China

Email: zhouyz@nju.edu.cn

========================README.TXT=========================