Constructing high-dimensional neural network potentials: CENT1 Method

Ehsan Rahmatizad

2^{nd} Workshop on Machine Learning in physics:
Applications in Condensed Matter Physics

03-05 Oct. 2018

Overview

1. Construction of the CENT1 NN Potential
   - Which files are needed for training the CENT1?
   - How to build a neural network architecture?
   - How to set initial parameters and do train?
   - Which of the trains are good?

2. How to use the NN potential?
   - Minhopp

Input files

training data set:

- input data:
  - posinp_train.yaml
  - posinp_valid.yaml
- `input files:`
  - ?.ann.input.yaml
  - flame_in.yaml
  - list_posinp_train.yaml (list of the data set for training)
  - list_posinp_valid.yaml (list of the data set for validating)

Neural network architecture

?.ann.input.yaml

- `nodes:`
  length of array shows number of hidden layers
  value of each item shows number of nodes in each hidden layer
- `rcut:` cutoff radius in Å
- `ampl_chi:` is the α in the equation 1
- `prefactor_chi:` is the β in the equation 1
- `chi0:` is χ_i^0 in equation 1

$$\chi = \chi_i^0 + \alpha tanh(\beta \times outputofNN) \tag{1}$$

Neural network architecture

...

- `ener_ref`: Single atom energy
- `gausswidth`: is α_i in equation 3
- `hardness`: is J_{ii} in equation 2

$$U_{tot}(\{q_i\}) = \sum_{i=1}^{N} (E_i^0 + \chi_i q_i + \frac{1}{2} J_{ii} q_i^2) + \frac{1}{2} \int \int \frac{\rho(r)\rho(r')}{|r - r'|} dr dr' \quad (2)$$

$$\rho_i(r) = \frac{q_i}{\alpha_i^3 \pi^{\frac{3}{2}}} e^{(-\frac{|r - r_i|^2}{\alpha_i^2})} \quad (3)$$

Somayeh Faraji, et al.,Phys. Rev. B 95, 104105 (2017)

Neural network architecture

...

- `qinit`: initial charge of the ion
- `method`: method of the calculation of the symmetry function values
- `symfunc`: determine type, number and parameters of symmetry functions
  types (implemented in FLAME) : g02 and g05

  J. Behler, JCP 134, 074106 (2011)

Initial parameters and train

flame_in.yaml

```yaml
main:
    task :  ann
    seed :  SEED
    types :  Na Cl
    verbosity :  2
ann:
    subtask :  train
    optimizer :  rivals
    approach :  cent1
    nstep_ekf :  12
    nconf_rmse :  400
    ampl_rand :  AMPL
    symfunc :  only_calculate
    print_energy :  False
```

Initial parameters and train

flame_in.yaml

```
potential:
    potential :ann
    print_force :  True
    ewald:
        ewald :  False
```

Example

- Minhopp

Minhopp

- input data:
  posinp.yaml
- `input files:`
- ?.ann.param.yaml: output potentials of CENT1
- input.minhopp: for setting edif, ekin, dt
- input.minhopp.bak: for clean.sh
- earr.dat : number of structures accepted and their energy.
- flame_in.yaml

S. Goedecker, JCP 120, 9911 (2004)

Initial parameters for minhopp

flame_in.yaml


```yaml
main:
  task: minhopp
  types: Na Cl
  verbosity : 1
  two_level_geopt: True
ann:
  approach: cent1
  syslinsolver : operator
```

flame_in.yaml


```yaml
potential:
  potential: ann
  print_force : True
  ewald:
        ewald: True
        psolver: p3d
        cell_ortho: True
        ecut: 3.0
        ecutz: 4.0
        rgcut: 5.0
        alpha: 2.0
```

flame_in.yaml


```yaml
geopt:
  method:  SQNM
  #method:  FIRE
  fmaxtol:  1.E-5
  condnum:  100.0
  alphax:  2.0
  precaution:  normal
  lprint:  True
geopt_prec:
  method:  SD
  fmaxtol:  1.E-1
  alphax:  0.6
  lprint:  True
```

flame_in.yaml


```yaml
minhopp:
  nstep: 150
  nsoften: 5
  mdmin: 3
  etoler: 2.E-4
  nrandoff: 5
  eref: -22405.290
  npminx: 5000
  trajectory: True
  print_force: True
  minter: 1
  ekinmax: 0.0150
```

COMMANDS

- train:

```
cd cent1
cd train
$FLAME_BUILD_DIR$/src/flame > o1 &
```

```
grep rmse train_output.yaml
```

- minhopp:

```
cd ..
cd minhopp
cp ../train/Cl.ann.param.yaml.00012 Cl.ann.param.yaml
cp ../train/Na.ann.param.yaml.00012 Na.ann.param.yaml
clean.sh
$FLAME_BUILD_DIR$/src/flame > o1 &
vim earr.dat
yaml2ascii.py poslow.yaml tt.ascii
v_sim
```

Thanks for your attention.