Constructing high-dimensional neural network potentials: Behler Method

Hossein Tahmasbi

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

03-05 Oct. 2018

Overview

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

2. How to use the NN potential?
   - Single point calculation
   - Geometry optimization

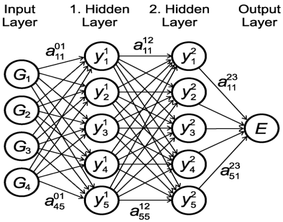
Input files

`/path/to/files/behler_method`

training data set: data (in yaml or binary format)
input files: `template_train`

- FLAME input file: `flame_in.yaml`
- input parameters for each element should be written in a seperate file: `B.ann.input.yaml` (e.g. for Boron)
- list of the data set for training: `list_posinp_train.yaml`
- list of the data set for validation: `list_posinp_valid.yaml`

Neural network architecture



B.ann.input.yaml

```yaml
main:
    nodes:  [5, 5]
    rcut:  9.0
    method:  behler
symfunc:
```

- `nodes:` number of hidden layers and their nodes
- `rcut:` cutoff radius in Å
- `method:` Behler symmetry functions
- `symfunc:` type, number and parameters of the symmetry functions
  types (implemented in FLAME) : g02 and g05

J. Behler, JCP 134, 074106 (2011)

Neural network architecture

g02 $\Rightarrow \eta,\ R_s = 0$

g05 $\Rightarrow \eta,\ \zeta,\ \lambda$

B.ann.input.yaml


```
symfunc:
    g02_001:  0.0010 0.0000 0.0000 0.0000 B
    g02_002:  0.0100 0.0000 0.0000 0.0000 B
    g02_003:  0.0200 0.0000 0.0000 0.0000 B
    :
    g05_001:  0.0001 1.000 -1.000 0.000 0.000 B B
    g05_002:  0.0001 1.700  1.000 0.000 0.000 B B
    g05_003:  0.0001 1.700 -1.000 0.000 0.000 B B
    :
```

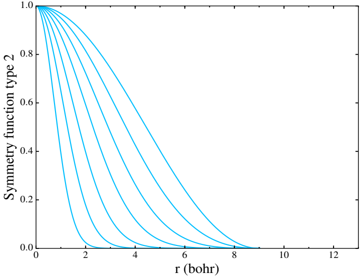
Neural network architecture

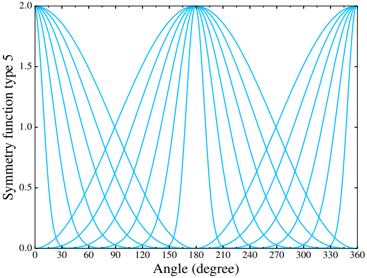
g02 $\Rightarrow \eta,\ R_s = 0$

$$G_i^2 = \sum_j e^{-\eta(R_{ij}-R_s)^2} . f_c(R_{ij})$$

g05 $\Rightarrow \eta,\ \zeta,\ \lambda$

$$G_i^5 = 2^{(1-\zeta)} \sum_{j,k\neq i}^{all} (1 + \lambda cos\theta_{ijk})^{\zeta}$$
$$e^{-\eta(R_{ij}^2+R_{ik}^2)} . f_c(R_{ij}) . f_c(R_{ik})$$





Initial parameters and train

flame_in.yaml


```yaml
main:
    task : ann
    seed : SEED
    types : B
    verbosity : 1
ann:
    subtask : train
    optimizer : rivals
    approach : atombased
    nstep_ekf : 20
    nconf_rmse : 1000
    ampl_rand : AMPL
    symfunc : only_calculate
    print_energy : False
```

Exercise

- Train
- Single point calculation
- Geometry optimization

Commands

- Train

  ```
  cp -r template_train test_train_01
  cd test_train_01
  ```

  check the input files and parameters and run flame

  ```
  /path/to/FLAME/src/flame & > o1&
  ```

- Single point calculation

  ```
  cp -r template_SP test_SP_01
  cp test_train_01/B.ann.param.yaml.???  test_SP_01/B.ann.param.yaml
  cd test_SP_01
  /path/to/FLAME/src/flame & > o1&
  ```

- Geometry optimization

  ```
  cp -r template_geopt test_geopt_01
  cp test_train_01/B.ann.param.yaml.???  test_geopt_01/B.ann.param.yaml
  cd test_geopt_01
  /path/to/FLAME/src/flame & > o1&
  ```

Good luck