A tutorial introduction to DeePMD-kit

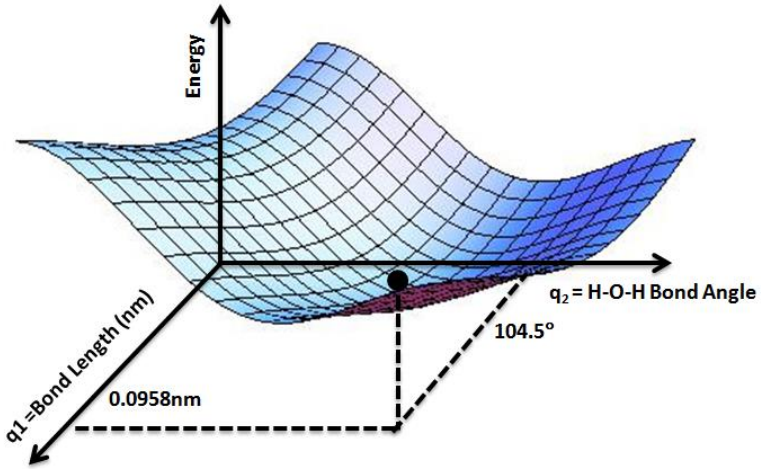
Xiaoyang Wang

Songshanhu Materials Lab
July 10 2020, DeePMD Workshop

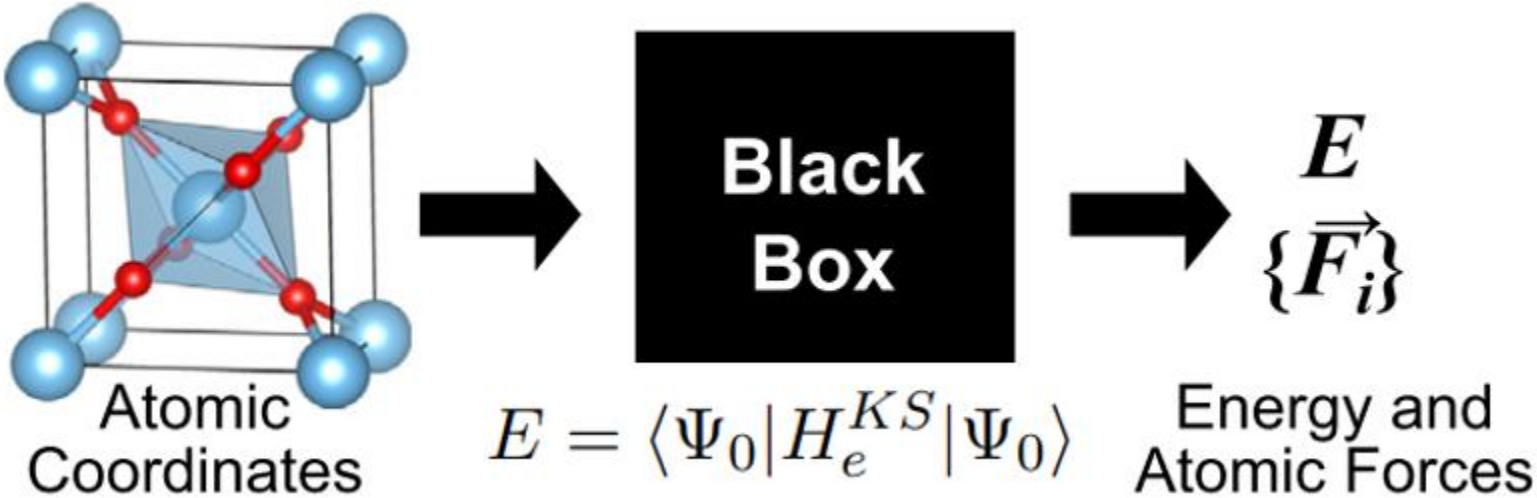
- Introduction
- Deep Potential
- DeepMD-kit

Classical molecular dynamics (MD)

$$m_i \frac{d^2 \boldsymbol{r}_i}{dt^2} = \boldsymbol{F}_i = -\nabla_{\boldsymbol{r}_i} E, \quad E = E(\boldsymbol{r}_1, ..., \boldsymbol{r}_i, ..., \boldsymbol{r}_N)$$



Accuracy v.s. efficiency dilemma



- First principle: accurate but very expensive.

  For example KS-DFT is limited to several hundred atoms.

Accuracy v.s. efficiency dilemma

- Empirical potentials: fast but limited accuracy.
  Lennard-Jones potential

$$V_{\mathrm{LJ}} = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] = \varepsilon\left[\left(\frac{r_{\mathrm{m}}}{r}\right)^{12} - 2\left(\frac{r_{\mathrm{m}}}{r}\right)^{6}\right]$$

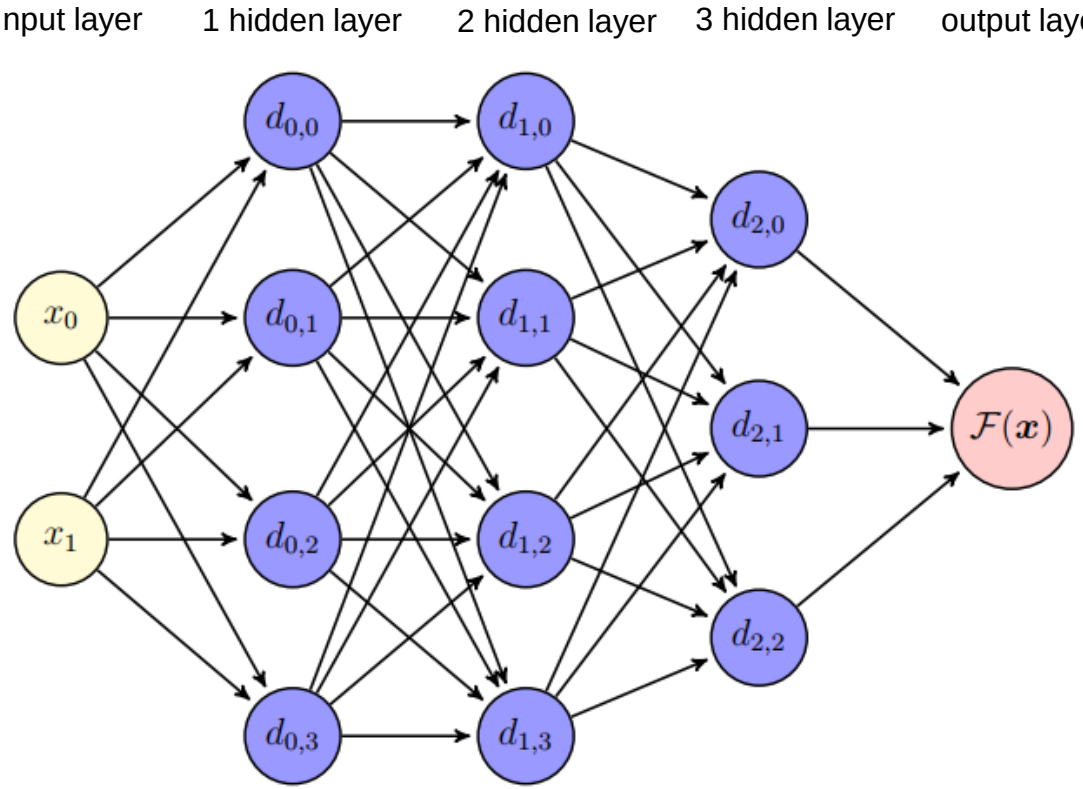
Morse potential

$$V(r) = D_e(1 - e^{-a(r-r_e)})^2$$

EAM potential （Tabulated in LAMMPS）

Deep Potential

Neural Network Potential

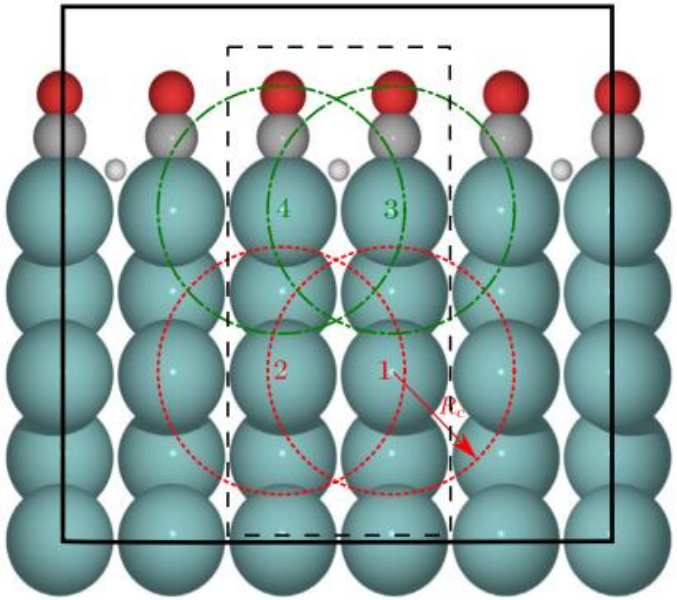


$$\boldsymbol{x} \xrightarrow{\mathcal{L}^0} \boldsymbol{d}_0 \xrightarrow{\mathcal{L}^1} \boldsymbol{d}_1 \xrightarrow{\mathcal{L}^2} \boldsymbol{d}_2 \xrightarrow{\mathcal{L}^{\text{out}}} \mathcal{F}(\boldsymbol{x})$$

$$\boldsymbol{d}^p = \mathcal{L}^p(\boldsymbol{d}^{p-1}) = \boldsymbol{d}^{p-1} + \Delta t\, \phi\Big(\boldsymbol{W}^p \cdot \boldsymbol{d}^{p-1} + \boldsymbol{b}^p\Big)$$

Atom-centered frame

Local approximation $$E = \sum_i E_i$$



$$\left\{(\tilde{\mathbf{R}}_1, E_1), (\tilde{\mathbf{R}}_2, E_2), \ldots\right\} \xrightarrow{\text{feature map and regression}} E = \sum_{i=1}^{N} \hat{E}_i(\mathbf{G}_i(\tilde{\mathbf{R}}^{(\mathrm{loc})}$$

Descriptor:

Gaussian

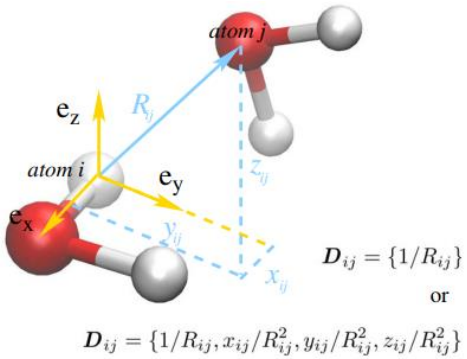
$$G_i^I = \sum_{j \neq i}^{\substack{\text{atoms j within } R_c \\ \text{distance of atom i}}} e^{-\eta (R_{ij} - R_s)^2 / R_c^2} f_c\left(R_{ij}\right)$$

Zernike

$$\rho_i(\mathbf{r}) = \sum_{j \neq i}^{\substack{\text{atoms j within } R_c \\ \text{distance of atom i}}} \eta_j \delta\left(\mathbf{r} - \mathbf{R}_{ij}\right) f_c\left(\|\mathbf{R}_{ij}\|\right)$$

Bispectrum -

DeepMD



https://amp.readthedocs.io/en/latest/theory.html

Descriptors: a smooth descriptor by DNN

local environment of atom *i* in terms

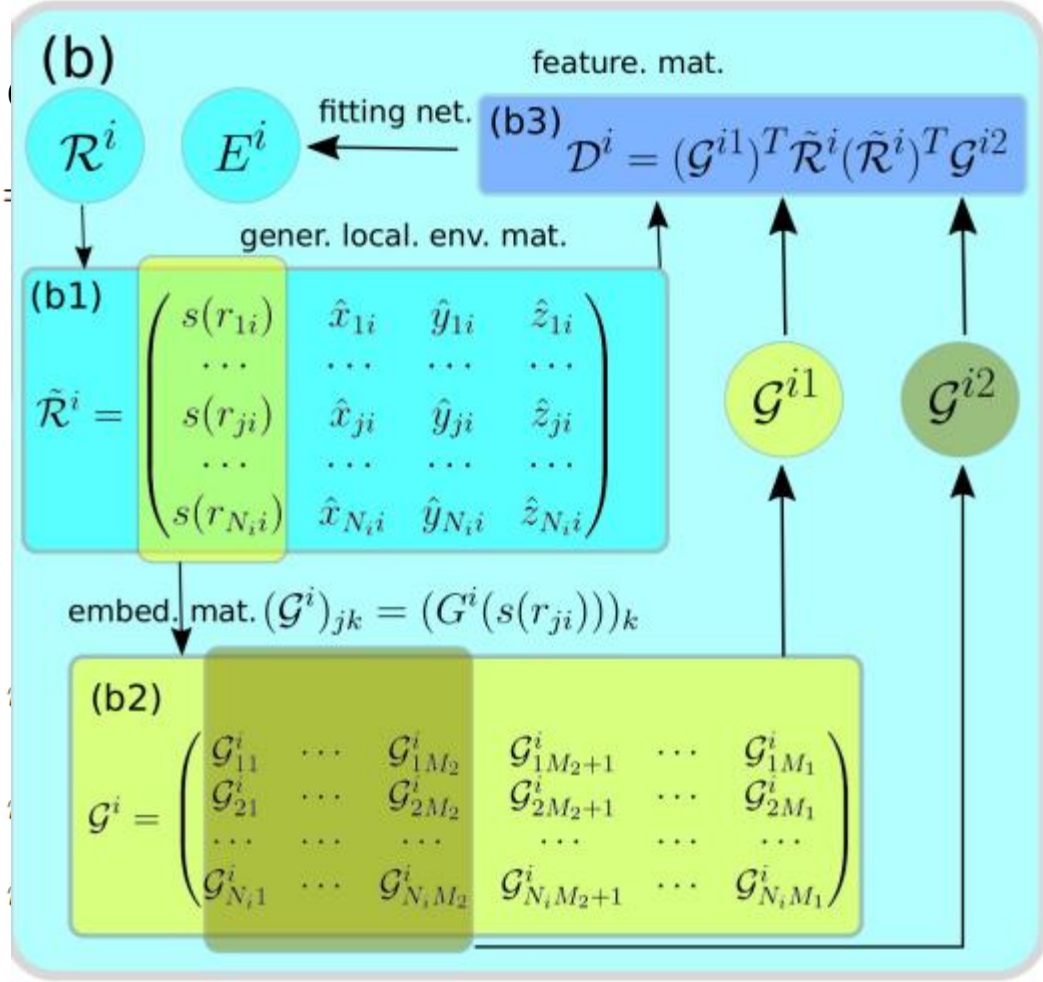
$$\mathcal{R}^i = \{\boldsymbol{r}_{1i}^T, \cdots, \boldsymbol{r}_{ji}^T, \cdots, \boldsymbol{r}_{N_i,i}^T\}^T, \ \boldsymbol{r}_{ji} =$$

Define generalized coordinates

$$\{x_{ji}, y_{ji}, z_{ji}\} \mapsto \{s(r_{ji}), \hat{x}_{ji}, \hat{y}_{ji}, \hat{z}_{ji}\}$$

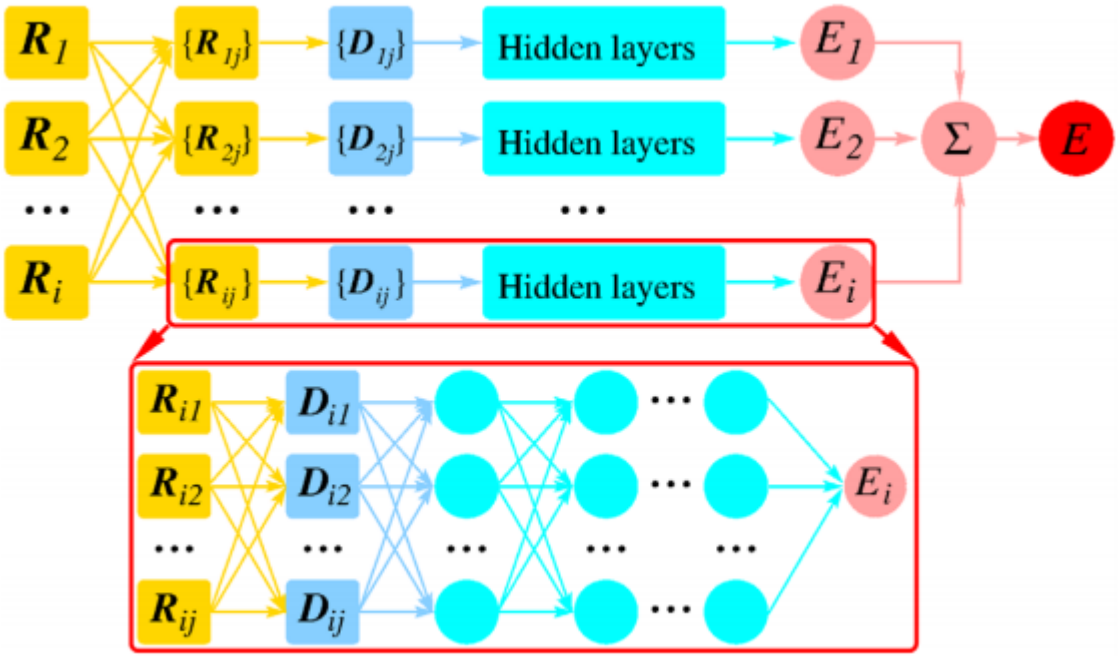
$$\hat{x}_{ji} = \frac{s(r_{ji})x_{ji}}{r_{ji}}, \ \hat{y}_{ji} = \frac{s(r_{ji})y_{ji}}{r_{ji}}, \ \hat{z}_{ji} =$$

$$s(r_{ji}) = \begin{cases} \frac{1}{r_{ji}}, \\ \frac{1}{r_{ji}}\left\{\frac{1}{2}\cos\left[\pi\frac{(r_{ji}-r_{cs})}{(r_c-r_{cs})}\right]+\frac{1}{2}\right\}, \\ 0, \end{cases}$$



Zhang, et.al., accepted by NIPS, 2018 Linfeng

The Deep Potential



https://github.com/deepmodeling/deepmd-kit

Deep potential: Training

Energy :

$$E = \sum_i \mathcal{N}_{\alpha_i}\left(\mathcal{D}_{\alpha_i}(r_i, \{r_j\}_{j\in n(i)})\right)$$

{[S1,E1,F1],
[S2,E2,F2],
…
[Sn,En,Fn]}

Force :

$$\boldsymbol{F}_i = -\nabla_{r_i} E$$

Loss function:

$$L(p_\epsilon, p_f, p_\xi) = \frac{p_\epsilon}{N}\Delta E^2 + \frac{p_f}{3N}\sum_i |\Delta \boldsymbol{F}_i|^2 + \frac{p_\xi}{9N}\|\Delta\Xi\|^2$$

where

$$p(t) = p^{\text{limit}}\left[1 - \frac{r_l(t)}{r_l^0}\right] + p^{\text{start}}\left[\frac{r_l(t)}{r_l^0}\right]$$

w weights in DNN. p_e and p_f are adaptively selected during the training process.

DeepMD-kit

DeepMD-kit

- DeepMD version and installation
- Prepare input file
- Train a model
- Freeze a model
- Test a model
- Model interfaces
- Run lammps with a model

DeepMD-kit version and installation

* Use conda

```
conda install deepmd-kit=*=*cpu lammps-dp=*=*cpu -c deepmodeling
```

```
conda install deepmd-kit=*=*gpu lammps-dp=*=*gpu -c deepmodeling
```

* Use offline package

deepmd-kit-1.2.0-cpu-Linux-x86_64.sh	435 MB
deepmd-kit-1.2.0-cuda10.1_gpu-Linux-x86_64.sh	1.29 GB

* Use source code

Source code (zip)

Source code (tar.gz)

https://github.com/deepmodeling/deepmd-kit#download-and-install

DeepMD-kit command

```
usage: dp [-h] {train,freeze,test} ...

DeepMD-kit: A deep learning package for many-body potential energy
representation and molecular dynamics

optional arguments:
  -h, --help            show this help message and exit

Valid subcommands:
  {train,freeze,test}
    train               train a model
    freeze              freeze the model
    test                test the model
```

```
#!/bin/sh
#SBATCH --partition=all
#SBATCH --job-name=dpdp
#SBATCH --mem=32G
#SBATCH --gres=gpu:1
#SBATCH --exclude gpu06,gpu07,gpu12
[redacted]

dp train water.json > runlog
```

Running command

Input file [water.json]

Water using se_a descriptor

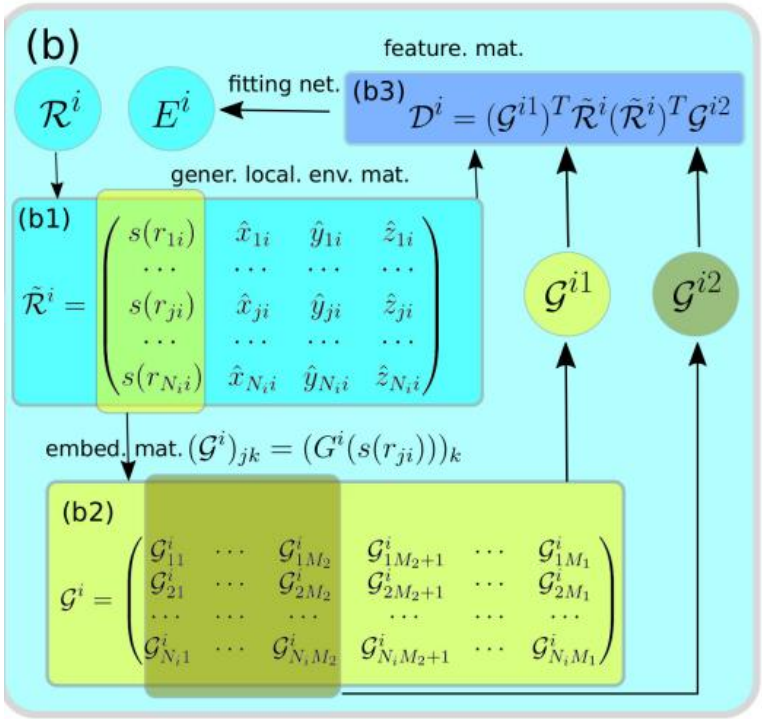
```
{
    "model": {
        "type_map":     ["O", "H"],
        "descriptor" :{
            "type":             "se_a",
            "sel":              [46, 92],
            "rcut_smth":        5.80,
            "rcut":             6.00,
            "neuron":           [25, 50, 100],
            "resnet_dt":        false,
            "axis_neuron":      16,
            "seed":             1
        },
        "fitting_net" : {
            "neuron":           [240, 240, 240],
            "resnet_dt":        true,
            "seed":             1
        }
    },

    "learning_rate" :{
        "type":         "exp",
        "start_lr":     0.001,
        "decay_steps":  2000,
        "decay_rate":   0.95
    },

    "loss" :{
        "start_pref_e": 0.02,
        "limit_pref_e": 1,
        "start_pref_f": 1000,
        "limit_pref_f": 1,
        "start_pref_v": 0,
        "limit_pref_v": 0
    },

    "training" : {
        "systems":      ["../../data/deepmd"],
        "set_prefix":   "set",
        "stop_batch":   400000,
        "batch_size":   1,
        "seed":         1,
        "disp_file":    "lcurve.out",
        "disp_freq":    100,
        "numb_test":    10,
        "save_freq":    1000,
        "save_ckpt":    "model.ckpt",
        "load_ckpt":    "model.ckpt",
        "disp_training":true,
        "time_training":true,
        "profiling":    false,
        "profiling_file":"timeline.json",
        "_comment":     "that's all"
```

Descriptor and fitting net

```
"descriptor" :{
    "type":             "se_a",
    "sel":              [46, 92],
    "rcut_smth":        5.80,
    "rcut":             6.00,
    "neuron":           [25, 50, 100],
    "resnet_dt":        false,
    "axis_neuron":      16,
    "seed":             1
```

$$s(r_{ji}) = \begin{cases} \dfrac{1}{r_{ji}}, & r_{ji} < r_{cs}. \\ \dfrac{1}{r_{ji}}\left\{\dfrac{1}{2}\cos\left[\pi\dfrac{(r_{ji}-r_{cs})}{(r_c-r_{cs})}\right]+\dfrac{1}{2}\right\}, & r_{cs} < r_{ji} < r_c. \\ 0, & r_{ji} > r_c. \end{cases}$$

```
"fitting_net" : {
    "neuron":           [240, 240, 240],
    "resnet_dt":        true,
    "seed":             1
}
```



Loss function and Learning rate

Loss function:

$$L(p_\epsilon, p_f, p_\xi) = \frac{p_\epsilon}{N}\Delta E^2 + \frac{p_f}{3N}\sum_i |\Delta \boldsymbol{F}_i|^2 + \frac{p_\xi}{9N}\|\Delta\Xi\|^2$$

```
"loss" :{
    "start_pref_e": 0.02,
    "limit_pref_e": 1,
    "start_pref_f": 1000,
    "limit_pref_f": 1,
    "start_pref_v": 0,
    "limit_pref_v": 0
},
```

Learning rate

$$p(t) = p^{\text{limit}}\left[1 - \frac{r_l(t)}{r_l^0}\right] + p^{\text{start}}\left[\frac{r_l(t)}{r_l^0}\right]$$

```
"learning_rate" :{
    "type":          "exp",
    "start_lr":      0.001,
    "decay_steps":   2000,
    "decay_rate":    0.95
},
```

Prepare data: raw format

For example: H-O potential project

system:
H2O, HO, O2, H2, H2O-300K, H2O-500K

frame:
A snapshot of a system that contains .raw information is called a frame
(Data with same element and number of atom)

box.raw, coord.raw, force.raw,
energy.raw and virial.raw, type.raw

Property	Unit
Time	ps
Length	Å
Energy	eV
Force	eV/Å
Pressure	Bar

Prepare data: raw format

Assuming system is O-H, we run 3 steps MD.
Then this system has 3 frames of data, every frame has 2 atoms

box.raw	3x9
coord.raw	3x6
force.raw	3x6
energy.raw	3x1
virial.raw	3x9
type.raw	2x1
type_map.raw	number_species x 1

```
$ cat force.raw
-0.724  2.039 -0.951  0.841 -0.464  0.363
 6.737  1.554 -5.587 -2.803  0.062  2.222
-1.968 -0.163  1.020 -0.225 -0.789  0.343
```

Prepare data: numpy format

```
deepmd/
├── box.raw
├── coord.raw
├── energy.raw
├── force.raw
├── set.000
│   ├── box.npy
│   ├── coord.npy
│   ├── energy.npy
│   └── force.npy
├── set.001
│   ├── box.npy
│   ├── coord.npy
│   ├── energy.npy
│   └── force.npy
├── type_map.raw
└── type.raw
```

Water project with 2 sets

The last set (set.001) is used as testing set, while the rest sets (set.000) are used as training sets

Prepare data: dpdata

a python package for manipulating DeePMD-kit, VASP, LAMMPS data formats.

Base class

System

Derived class

LabeledSystem

MultiSystem

key	type	dimension	are labels	description
'atom_names'	list of str	ntypes	False	The name of each atom type
'atom_numbs'	list of int	ntypes	False	The number of atoms of each atom type
'atom_types'	np.ndarray	natoms	False	Array assigning type to each atom
'cells'	np.ndarray	nframes x 3 x 3	False	The cell tensor of each frame
'coords'	np.ndarray	nframes x natoms x 3	False	The atom coordinates
'energies'	np.ndarray	nframes	True	The frame energies
'forces'	np.ndarray	nframes x natoms x 3	True	The atom forces
'virials'	np.ndarray	nframes x 3 x 3	True	The virial tensor of each frame

Prepare data: dpdata

Supported data format

Software	format	multi frames	labeled	class	format key
vasp	poscar	False	False	System	'vasp/poscar'
vasp	outcar	True	True	LabeledSystem	'vasp/outcar'
vasp	xml	True	True	LabeledSystem	'vasp/xml'
lammps	lmp	False	False	System	'lammps/lmp'
lammps	dump	True	False	System	'lammps/dump'
deepmd	raw	True	False	System	'deepmd/raw'
deepmd	npy	True	False	System	'deepmd/npy'
deepmd	raw	True	True	LabeledSystem	'deepmd/raw'
deepmd	npy	True	True	LabeledSystem	'deepmd/npy'
gaussian	log	False	True	LabeledSystem	'gaussian/log'
gaussian	log	True	True	LabeledSystem	'gaussian/md'
siesta	output	False	True	LabeledSystem	'siesta/output'
siesta	aimd_output	True	True	LabeledSystem	'siesta/aimd_output'
cp2k	output	False	True	LabeledSystem	'cp2k/output'
QE	log	False	True	LabeledSystem	'qe/pw/scf'
QE	log	True	False	System	'qe/cp/traj'
QE	log	True	True	LabeledSystem	'qe/cp/traj'

to_deepmd_raw

to_deepmd_npy

```
deepmd/
├── box.raw
├── coord.raw
├── energy.raw
├── force.raw
├── set.000
│   ├── box.npy
│   ├── coord.npy
│   ├── energy.npy
│   └── force.npy
├── set.001
│   ├── box.npy
│   ├── coord.npy
│   ├── energy.npy
│   └── force.npy
├── type_map.raw
└── type.raw
```

Train a model : output

command:

```
train                   train a model
freeze                  freeze the model
test                    test the model
```

nohup dp train water.json 1>runlog 2> err &

```
-rw-r--r-- 1 anguse pku4p     165 Jan 16 15:53 checkpoint
-rw-r--r-- 1 anguse pku4p   65535 Jan 16 15:53 lcurve.out
-rw-r--r-- 1 anguse pku4p 5861440 Jan 16 15:53 model.ckpt.data-00000-of-00001
-rw-r--r-- 1 anguse pku4p    4457 Jan 16 15:53 model.ckpt.index
-rw-r--r-- 1 anguse pku4p 1000914 Jan 16 15:53 model.ckpt.meta
-rw-r--r-- 1 anguse pku4p     192 Jan 16 15:08 _rsub
-rw-r--r-- 1 anguse pku4p   55545 Jan 16 15:53 runlog
-rw-r--r-- 1 anguse pku4p   25558 Jan 16 15:09 slurm-64773.out
-rw-r--r-- 1 anguse pku4p    1353 Jan 16 15:08 water.json
```

checkpoint	used for restarting
model.ckpt*	model related files
runlog	standard output (version, data, hardware info., time)
lcurve.out	statistic data (loss function info.)

Train a model : output file lcurve.out

```
(base) [root@iZ2ze5x8isyf7vkwoxkq6sZ ref]# head lcurve.out
# batch      l2_tst    l2_trn    l2_e_tst  l2_e_trn    l2_f_tst  l2_f_trn         lr
      0    3.25e+01  3.23e+01    1.03e+01  1.03e+01    8.08e-01  8.01e-01    1.0e-03
    100    2.59e+01  2.67e+01    1.71e+00  1.70e+00    8.13e-01  8.39e-01    1.0e-03
    200    2.54e+01  2.59e+01    2.25e-01  2.29e-01    8.03e-01  8.19e-01    1.0e-03
    300    2.44e+01  2.30e+01    1.55e-01  1.55e-01    7.72e-01  7.27e-01    1.0e-03
    400    2.21e+01  2.19e+01    3.00e-01  3.08e-01    6.98e-01  6.93e-01    1.0e-03
    500    2.05e+01  1.94e+01    1.71e-01  1.76e-01    6.48e-01  6.14e-01    1.0e-03
    600    1.46e+01  1.49e+01    1.42e-01  1.37e-01    4.61e-01  4.70e-01    1.0e-03
    700    1.22e+01  1.19e+01    1.31e-01  1.32e-01    3.85e-01  3.75e-01    1.0e-03
    800    1.35e+01  1.35e+01    3.74e-02  4.24e-02    4.28e-01  4.28e-01    1.0e-03
(base) [root@iZ2ze5x8isyf7vkwoxkq6sZ ref]# tail lcurve.out
 399100    4.76e-02  4.46e-02    5.27e-04  1.54e-04    4.62e-02  4.37e-02    3.7e-08
 399200    4.76e-02  4.82e-02    5.13e-04  1.87e-04    4.62e-02  4.73e-02    3.7e-08
 399300    4.76e-02  4.24e-02    5.10e-04  1.19e-04    4.62e-02  4.16e-02    3.7e-08
 399400    4.75e-02  4.39e-02    4.93e-04  4.12e-04    4.62e-02  4.27e-02    3.7e-08
 399500    4.76e-02  4.42e-02    5.24e-04  5.64e-04    4.62e-02  4.28e-02    3.7e-08
 399600    4.75e-02  4.13e-02    5.03e-04  3.76e-06    4.62e-02  4.05e-02    3.7e-08
 399700    4.76e-02  4.24e-02    5.29e-04  1.33e-04    4.62e-02  4.16e-02    3.7e-08
 399800    4.76e-02  4.62e-02    5.09e-04  3.05e-06    4.62e-02  4.54e-02    3.7e-08
 399900    4.76e-02  4.63e-02    5.40e-04  1.07e-04    4.62e-02  4.54e-02    3.7e-08
 400000    4.75e-02  4.62e-02    5.22e-04  1.40e-04    4.62e-02  4.54e-02    3.5e-08
```

l2_tst l2_trn	total error of test sets and training sets
l2_e_tst l2_e_trn	energy error of test sets and training sets
l2_f_tst l2_f_trn	force error of test sets and training sets

$$L_2(\mathrm{E}) = \left|\left|\boldsymbol{E}_{predicted} - \boldsymbol{E}_{test}\right|\right|_2$$

Freeze a model

command:

dp freeze — Model file: frozen_model.pb

dp freeze -o graph.pb — Model file: graph.pb

```
train                   train a model
freeze                  freeze the model
test                    test the model
```

Test a model

command:

dp test -m graph.pb -s /root/workshop/deepmd-kit/data/test/ -d result

dp test -m graph.pb -s /root/workshop/deepmd-kit/data/ -d result **Latest release

```
train                   train a model
freeze                  freeze the model
test                    test the model
```

-m Model file graph.pb
-s test set directory
-d save detail info. about energy, force and viral

```
test/
├── set.000
│   ├── box.npy
│   ├── coord.npy
│   ├── energy.npy
│   └── force.npy
├── type_map.raw
└── type.raw
```

```
# number of test data : 30
Energy L2err        : 7.709832e-02 eV
Energy L2err/Natoms : 4.015538e-04 eV
Force  L2err        : 4.488686e-02 eV/A
Virial L2err        : 4.900048e+00 eV
Virial L2err/Natoms : 2.552108e-02 eV
```

```
result.e.out  result.f.out  result.v.out
```

-m Model file graph.pb
-s test set directory
-d save detail info. about energy, force and viral

Quality of the model: test results, lcurve.out

Model python interface

```
import deepmd.DeepPot as DP
from pprint import pprint
import numpy as np
dp = DP('graph.pb')
coord = np.array([[1,0,0], [0,0,1.5], [1,0,3]]).reshape([1, -1])
cell = np.diag(10 * np.ones(3)).reshape([1, -1])
atype = [1,0,1]
e, f, v = dp.eval(coord, cell, atype)
print('-'*20)
pprint(e)
print('-'*20)
pprint(f)
print('-'*20)
pprint(v)
```

run.py

```
--------------------
array([[-463.85127894]])
--------------------
array([[[-0.57670531,  0.        ,  0.78576773],
        [ 1.15341063,  0.        ,  0.        ],
        [-0.57670531,  0.        , -0.78576773]]])
--------------------
array([[-1.15341063,  0.        ,  0.        ,  0.        ,  0.        ,
         0.        ,  0.        ,  0.        , -2.3573032 ]])
~
~
~
~
```

runlog [+]

Model ASE interface

```
from ase import Atoms
from pprint import pprint
from deepmd.calculator import DP
water = Atoms('H2O',
              positions=[(0.7601, 1.9270, 1),
                         (1.9575, 1, 1),
                         (1., 1., 1.)],
              cell=[100, 100, 100],
              calculator=DP(model="graph.pb"))
pprint(water.get_potential_energy())
pprint(water.get_forces())
```

run.py

```
array([-467.32403965])
array([[-0.24308831,  0.15358002,  0.        ],
       [ 0.21159887, -0.19686582,  0.        ],
       [ 0.03148944,  0.0432858 ,  0.        ]])
```

run.log [+]

MD runs with the model

Simple water example

```
units           metal
boundary        p p p
atom_style      atomic

neighbor        2.0 bin
neigh_modify    every 10 delay 0 check no

read_data       water.lmp
mass            1 16
mass            2 2

pair_style      deepmd frozen_model.pb
pair_coeff

velocity        all create 330.0 23456789

fix             1 all nvt temp 330.0 330.0 0.5
timestep        0.0005
thermo_style    custom step pe ke etotal temp press vol
thermo          100
dump            1 all custom 100 water.dump id type x y z

run             1000
```

New Features in latest release

New features of dp train:

- Polarizability and dipole fitting
- If provided with stop_lr, the decay_rate will be computed automatically
- Support non-pbc system: add an empty file named nopbc to the data system.
- Use envs TF_INTRA_OP_PARALLELISM_THREADS and TF_INTER_OP_PARALLELISM_THREADS to control the multi-threading of tf, clean up command line options.
- When the key systems is provided with a string, all possible systems will be recursively searched within the dir given by systems
- User specific atomic energy.
- Exclude types when building descriptors.
- User specific activation function and network precision.
- Transform neural network parameters.

New features of dp test:

- dp test -s system also support recursive dir searching
- Weighted average of test results

Maximum neighborlist size =1024.

DeepMD-kit

If provided with stop_lr, the decay_rate will be computed automatically

```
"learning_rate" :{
    "type":         "exp",
    "start_lr":     0.001,
    "decay_steps":  2000,
    "decay_rate":   0.95
},
```

Support non-pbc system: add an empty file named nopbc to the data system.

```
(base) [root@iZ2zeg67k0g35ak5ypgcgnZ deepmd]# ls
box.raw  coord.raw  energy.raw  force.raw  nopbc  set.000  set.001  type_map.raw  type.raw
(base) [root@iZ2zeg67k0g35ak5ypgcgnZ deepmd]#
```

DeepMD-kit

Use envs TF_INTRA_OP_PARALLELISM_THREADS and TF_INTER_OP_PARALLELISM_THREADS to control the multi-threading of Tensorflow, clean up command l ine options.

```
// The execution of an individual op (for some op types) can be
// parallelized on a pool of intra_op_parallelism_threads.
```

Recursively search all possible systems in a directory.

"systems": ["../../data/1/deepmd", "../../data/2/deepmd", "../../data/3/deepmd"]

"systems": ["../../data/"] ⟶ automatic search all sub-folders for systems

dp test -s system also support recursive dir searching

command:

dp test -m graph.pb -s /root/workshop/deepmd-kit/data/test/ -d result

dp test -m graph.pb -s /root/workshop/deepmd-kit/data/ -d result **Latest release

Acknowledgements

* Songshan Lake Material Lab.
* Alibaba Cloud Computing

* Weinan E , Linfeng Zhang (Princeton Univ.)
* Han Wang (IAPCM)
* Yuzhi Zhang, Weijie Chen, Fengbo Yuan(Peking Univ.)
* Wanrun Jiang and Zhaohan Ding
* Jianxing Huang (Xiamen Univ.)
* Jinzhe Zeng (ECNU)

Thank you !

http://www.deepmd.org/
https://github.com/deepmodeling/