A1: Interpretability of descriptors in DeePMD-kit

Qiang Zhu, Yuming Gu, Xinzhu Wang, Limu Hu

2021-08-09

Contents

A1: Interpretability of descriptors in DeePMD-kit

1 Discriptions of 6 given systems with the raw data

1.1 Scripts converting raw data to xyz for visualization

Here we provide a package named raw2pdb which could convert the ***raw*** data file to ***exyz*** (an extended xyz format) and ***pdb*** format. The ***pbc*** information was also encoded into these two type files. You can visualize 3D structures with VMD or PyMol.
An example code is show below:

```python
from raw2pdb import raw2pdb

# path to the raw data
idir = '/Users/zhuqiang/Documents/My_Jobs@Nanjing/My_Competition/
    deepmd_hackathon/Cu_full/cu.bcc.02x02x02/02.md/sys-0016/'deepmd
# path to the output dir
odir = '.'
# output name
oname = 'cu.bcc.02x02x02.pdb'
# execute the func
raw2pdb(idir,odir,oname)
```

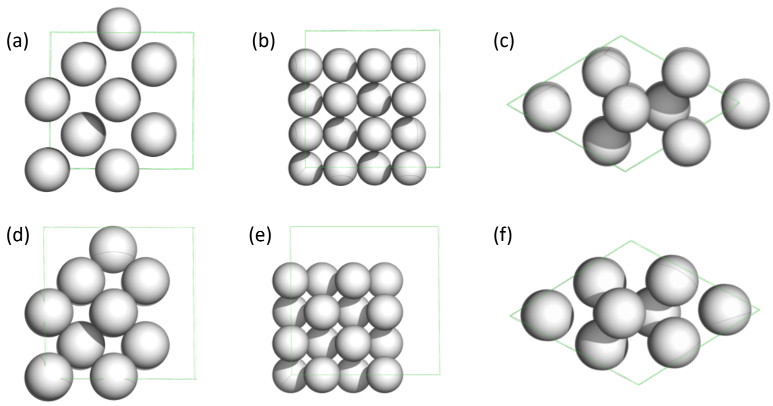

Figure 1: Illustration of the raw data of 6 given systems, namely, Cu (a) *bcc*, (b) *fcc*, (c) *hcp* under normal conditions and (d) *bcc*, (e) *fcc*, (f) *hcp* under high pressure

Once you get the ***pdb*** or ***xyz*** file, you can visualize it with some state-of-art software. PyMol was utilized here.

1.2 Visualization of the feature spaces

Here we revised two scripts of the code for dumping the feature spaces.

```python
#--- se_a.py ---#
        tf.add_to_collection('coutput',output)
        return output, output_qmat

#--- trainer.py ---#
                doutput,_ = run_sess(self.sess, [tf.get_collection('coutput'),
    self.train_op], \
                feed_dict=train_feed_dict,\
                options=prf_options, run_metadata=prf_run_metadata)

                doutput = np.array(doutput)
                # dump the feature matrix to doutput.dat
                if cur_batch %1000 == 0:
```

```python
                with open('./doutput.dat','a') as f:
                    for i in range(doutput.shape[0]):
                        for j in range(doutput.shape[1]):
                            for k in range(doutput.shape[2]):
                                f.write('{:.5f} '.format(float(doutput[i][j][k])))
                            f.write('\n')
```

With the feature matrix $\mathcal{D}_i$[1] possessed, we want to know the relationship between the feature space and the ***raw*** data we provided.

As the feature matrix has a dimensional of $M1 * M2$. In the Cu system, the parameter of $M1$ and $M2$ are specified by the last layer of the *neuron* and *axis_neuron*, respectively, which results dimension of 1200 for a single atom. It is too large to be visualize. The first task is to reduce the dimension. Here, we applied the principle component analysis (PCA) with help of the scikit-learn.[2]

As shown in Figure 2 (a), we plotted top 10 ranked principle components (pcs). The top 3 pcs could cover almost 93 %, which means 1200 dimensions could be reduced to 3 without lossing much information. Subsuquently, we further group the data into 6 groups with the K-Means algorithm supplied by sci-kit learn,[2] and the results were shown in Figure 2 (b).

Why do we choose 6 groups? As we firstly guess, the feature spaces could well seperate the 6 Cu systems. The results may somewhat be depressed.

How did the 6 systems look like? if we visualize it with some conventional ways. In the next subsection, the radial distribution functions were applied.

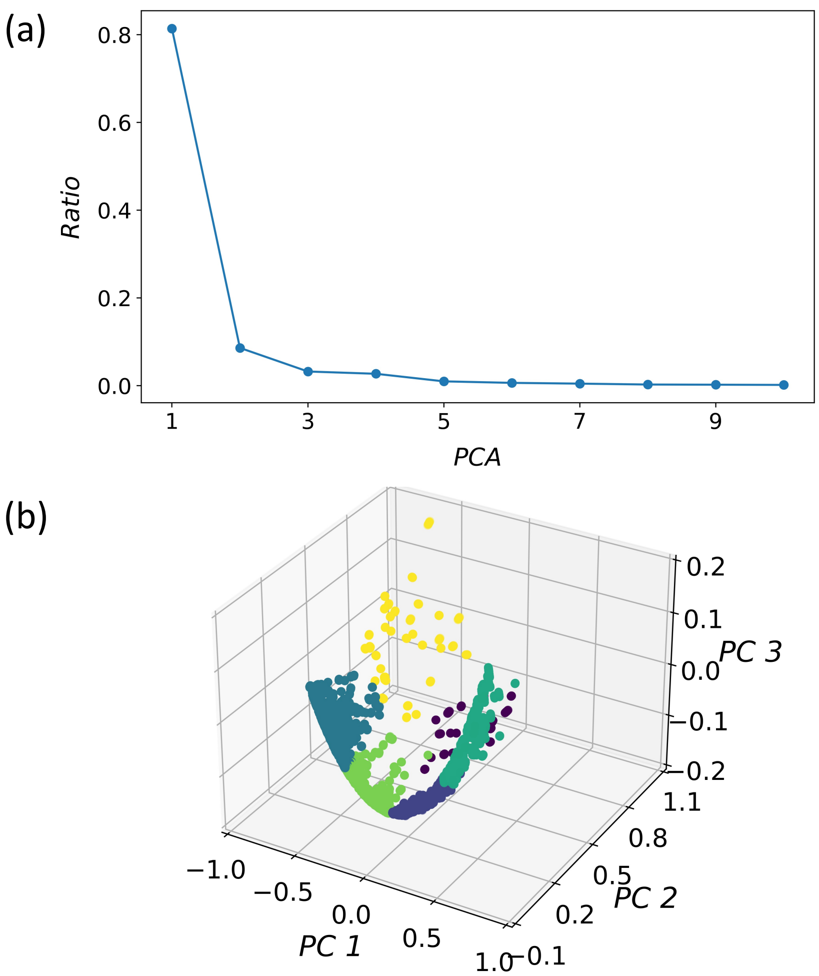

Figure 2: Visualization of the feature space. (a) Ratio of top 10 eigenvalues (b) Feature space of the feature matrix $\mathcal{D}_i$ mapped to the top 3 components and grouped into 6 clusters with KMeans.

1.3 Radial Distribution Functions

Here, the radial distribution functions (rdfs) of 6 Cu systems were calculated with MDTraj.[3] The results are shown in Figure 3. From it, we could see that 6 rdfs could be grouped into 2 large classes, where the first peak changed a lot. These two classes denote the systems under the normal condition (solid lines in Figure 3) and high pressure (dashed lines), respectively. A small shift of the first peak was observed in the *bcc*, *fcc*, and *hcp*. However, little difference could be observed within these two classes. This phenomenon may also be reflected from Figure 2 (b).

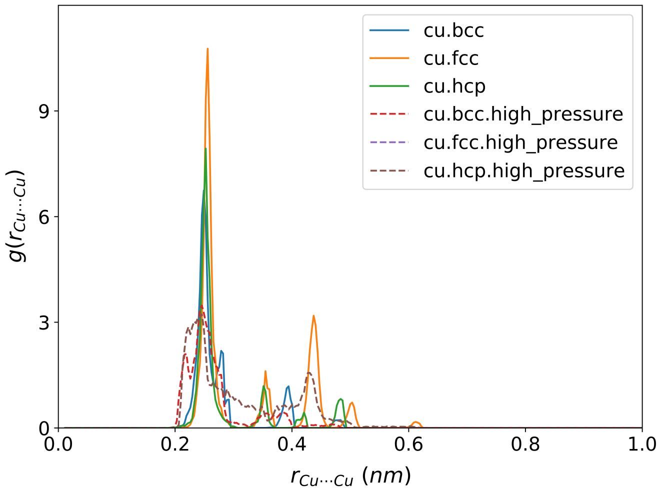

Figure 3: Radial distribution functions of 6 Cu systems.

References

[1] Linfeng Zhang, Jiequn Han, Han Wang, Wissam A Saidi, Roberto Car, et al. End-to-end symmetry preserving inter-atomic potential energy model for finite and extended systems. *arXiv preprint arXiv:1805.09003*, 2018.

[2] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[3] Robert T. McGibbon, Kyle A. Beauchamp, Matthew P. Harrigan, Christoph Klein, Jason M. Swails, Carlos X. Hernández, Christian R. Schwantes, Lee-Ping Wang, Thomas J. Lane, and Vijay S. Pande. Mdtraj: A modern open library for the analysis of molecular dynamics trajectories. *Biophysical Journal*, 109(8):1528 – 1532, 2015.