Due date: Wednesday, January 05, 2021

!!Please make sure that you properly comment your code in the jupyter notebook!!

1 Neural network for classification of bulk phases

In this worksheet, we develop a neural network (NN) for the classification of local structural environments in bulk phases. There will be three steps that will be worked on over the next three weeks:

(1) setting up the bulk phases and defining the descriptors (week 1)

(2) **optimizing the hyper-parameters of the descriptors (week 2)**

(3) training the NN and application (week 3)

Each step will be submitted separately and the solution will be presented in the tutorial, so that it will be possible to work on subsequent tasks even if the solution to the previous task could not be found.

2 Optimizing hyper-parameters of symmetry functions (20 pt)

The symmetry functions G_1, G_2, G_3 introduced in the last exercise depend on five parameters: r_{min} and r_c entering the cutoff function f_c, R_s and η in G_2, and κ in G_3. For the same local environment, the values of the symmetry functions strongly depend on the selected parameters. Here, we want to find the optimal set of parameters for classifying the four bulk phases fcc, bcc, hcp, and liquid.

In terms of classification, the n input descriptors to the NN span an n-dimensional space and by optimizing the weights of the NN, the decision boundaries between the different classes in this n-dimensional space are optimized. For computational efficiency, the number of input descriptors should be kept small. Each *type* of symmetry function (G_1, G_2, G_3) can enter several times with different values for the respective parameters. The best results are expected for parameters that give rather different values for the symmetry functions in the different environments.

At finite temperatures, thermal vibrations will induce fluctuations in the values of the symmetry functions for different atoms in the same structure. We therefore need to evaluate the distributions of symmetry function values for a given set of parameters in each of the four structures and compute the overlap between the distributions to assess the usefulness of different sets of parameters.

2.1 Task 1: Computing symmetry function values

For each of the four bulk structures, we will compute symmetry functions for a range of values and for a number of atoms from the MD trajectories. We will use the trajectories from the last exercise, that is fcc, bcc, and hcp at $T = 600$ K and liquid at $T = 3000$ K. Each trajectory

should contain 101 slices. The number of atoms in each slice should be 1372 for fcc, 1024 for bcc, 1134 for hcp, and 1000 for liquid, respectively.

We consider the first 50 slices of each trajectory as equilibration stage. For each bulk structure, read in the MD trajectory and randomly pick 10 slices between slice 50 and 101. From each slice randomly pick 100 atoms (different for each slice), yielding a total of 1000 atoms (local environments) for each structure. The parameters are varied as follows:

- $r_{\text{min}} = 2.6$, $r_c = 2.8$
- $R_s \in [1.0, 2.5]$ in steps of 0.1 (16 values in total), $\eta = [2.0, 10.0, 100.0, 800.0]$ (4 values in total)
- $\kappa \in [0.5, 10.5]$ in steps of 0.5 (21 values in total)

This means, that for each atom we need to compute 1 value for G_1, 64 values for G_2, and 21 values for G_3. If you collect the values in an array, you would have arrays of 1000×1 for G_1, 1000×64 for G_2, and 1000×21 for G_3. You can also use pickle to dump the data into a file and then read them in again later:

```
import pickle as pck

# dump to file
with open('g_fcc600.pck', 'wb') as f:
     pck.dump((g1_fcc600, g2_fcc600, g3_fcc600), f)

# read from file
with open('g_fcc600.pck', 'rb') as f:
    (g1_fcc600, g2_fcc600, g3_fcc600) = pck.load(f)
```

where `g1_fcc600`, `g2_fcc600`, and `g3_fcc600` are the corresponding arrays (here, for example, for fcc at $T = 600$ K).

2.2 Task 2: Overlap between histograms

To compare the symmetry function values for different structures with the same parameters, we compute the overlap between the distributions obtained from the 1000 randomly selected atoms. Create the histograms using `numpy.histogram` (https://numpy.org/doc/stable/reference/generated/numpy.histogram.html). We want normalized histograms, therefore we set `density = True`. Furthermore, the histograms for the same symmetry function with the same parameters need to have the same range for the different structures. For each set of parameters, find the minimum and maximum value for the histogram over *all* four structures and use those values to set the `range` of the histograms. These ranges can vary quite a bit. For instance, the G_3 function with $\kappa = 6.5$ has the following ranges of values:

- fcc: [0.98, 3.81]
- bcc: [12.79, 24.64]
- hcp: [10.05, 14.38]

- liquid: [-23.61, 11.40]

Consequently, the overall range that should be set for all structures for the G_3 function with $\kappa = 6.5$ is [-23.61, 24.64]. For other functions with other parameters, the values will be different. Set the number of bins to 30 in all cases.

Compute the overlap between histograms for each pair of structures, that is fcc–bcc, fcc–hcp, fcc–liq, bcc–hcp, bcc–liq, and hcp–liq. Make sure that you properly normalize with the bin width, such that 100% overlap results in a value of 1.0 and no overlap in 0.0, respectively.

2.3 Task 3: Visualization and parameter selection

We now want to assess which parameter sets could be most suitable for our classification task. For the G_1 function with r_{min} and r_c, report the overlap for each pair of structures. Which structures might already be well separated with this function? For G_2 and G_3, plot the magnitude of the overlap for each pair of structures as a function of the parameter set index ($0 - 63$ for G_2 and $0 - 20$ for G_3).

As you can see, there is no set of parameters for which the overlap between *all* pairs is zero. Therefore, we need to select several parameters, such that the overlap between any two structures is small at least once. Using the overlap as a guide, select 5 G_2 functions (5 combinations of R_s and η) and 5 G_3 functions (5 values of κ). Plot the distributions of the values for these symmetry functions for the four bulk structures (have a look at https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.hist.html for histogram plotting with matplotlib). Make a separate plot for each of the selected functions (1 for G_1, 5 for G_2, and 5 for G_3, so 11 plots in total). Use the same number of bins and the same ranges for plotting as for the computation of the histograms and plot the probability density (such that the integral is 1). Report the corresponding values of R_s and η or κ in the respective plots. If after visually inspecting the histograms, you are not satisfied with your choice of functions (not providing good separation between at least two structures) you can, of course, update your selection.

❄ ❄ ❄

Merry Christmas

And

A Happy New Year!

❄ ❄ ❄