Aim of the Project

The aim of the project is to develop a methodology to generate coarse grained pair potentials for molecular simulation, using the machinery of deep learning.

Overview

How the methodology works:

1. Atomistic simulation is run
2. Atomistic trajectory is coarse grained (using MagiC cgtraj)
3. Coarse grained RDF(s) are calculated from CG trajectory (using MagiC rdf.py)
4. The CG system is set up in LAMMPS (everything except potentials). The system should be in the same initial configuration and conditions as the atomistic simulation, but with CG beads instead of individual atoms
5. Simulations of the CG system are run using generated trial potentials (using LAMMPS currently) and RDFs are calculated from these
6. The set of RDFs and their corresponding potentials are used as training data to train a deep neural network, with RDFs as inputs and potentials as outputs The ‘forbidden region’, i.e. the region at short distances where the potential is very high and the corresponding RDF value is 0 is excluded for better network performance
7. The CG RDF(s) from (3) are input into the trained network, and the output potentials can now be used for a CG simulation

Steps 1-3 are identical to MagiC’s initial procedure

Trial Potentials

The potentials are defined in tables, and Python functions are used to go between MagiC, LAMMPS and numpy array formats.

When creating the trial potentials, it is important to have the size of the ‘forbidden region’ (i.e. the max value of the radius at which the potential is very large) matches the first non-zero region from the initial RDF. This is necessary as the RDFs and potentials will have this region excluded, and the training data needs to be the same length as the actual true data (This may become an issue when multiple bead types are present, and as such multiple RDFs with different lengths make up a single input).

The potentials are currently generated by a method where each point in the potential is found by taking the previous point and adding a value, where this value is found by taking a random value from a gaussian distribution centered at 0 and multiplying it by a random weight. The starting potential just outside the forbidden region is also given by a gaussian.

Within the forbidden region, the potential is very high, linearly decreasing.

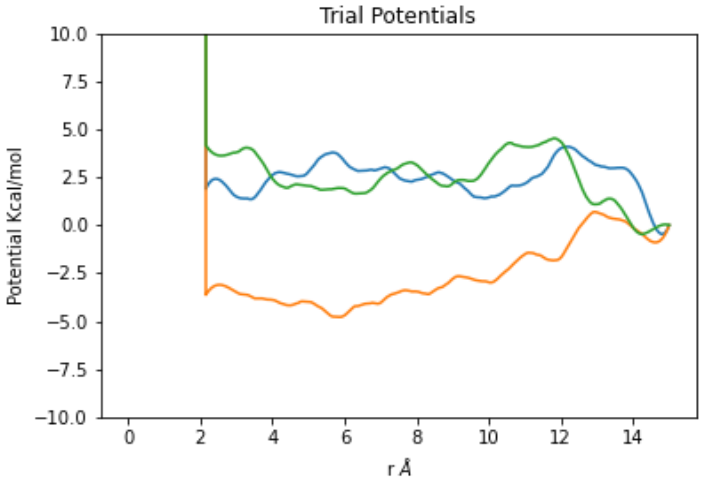

Fig. 1: 3 of the generated trial potentials

It is likely that better methods for generating potentials exist, which may be a possible area for improvement, perhaps incorporating Leonard-Jones or Coulomb potentials. For example most potentials tend to zero as they go to infinity with a gradual slope at larger distances, which is not represented by these potentials. Many real potentials also have a smaller number of larger features, whereas these created potentials have a high number of smaller features.

Example

The example system was 20 Na+ ions. The potential used was taken from a MagiC tutorial (Na-Cl tutorial), and the initial simulation was 10ns. 10,000 simulations were run using trial potentials (3ns each) from which RDFs were generated, and a neural network was trained using this data.

Results

The network was trained using ADAM optimizer and Mean Squared Error loss. The best performing network architecture was as follows:

```
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_13 (Dense)             (None, 257)               66306
_________________________________________________________________
dense_14 (Dense)             (None, 80)                20640
_________________________________________________________________
dense_15 (Dense)             (None, 80)                6480
_________________________________________________________________
dense_16 (Dense)             (None, 257)               20817
=================================================================
Total params: 114,243
Trainable params: 114,243
Non-trainable params: 0
_________________________________________________________________
```

Fig. 2: Network Architecture

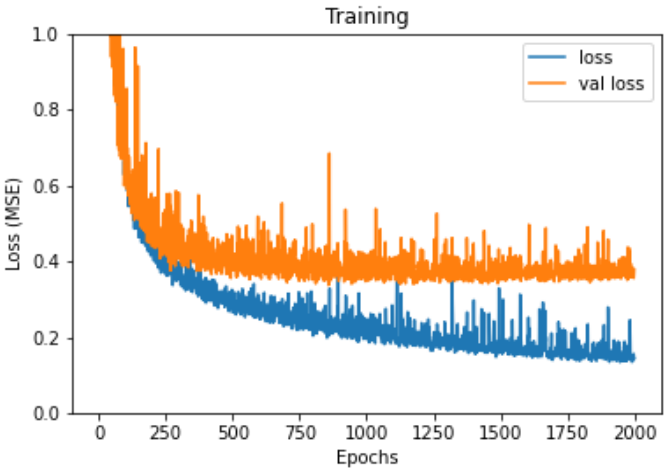

Fig. 3: Network Performance of sample network during Training

The (MSE) loss after training for ~2000 epochs was loss: 0.1921 - val_loss: 0.3682. When the Na-Na RDF was input into the trained network the following was obtained:

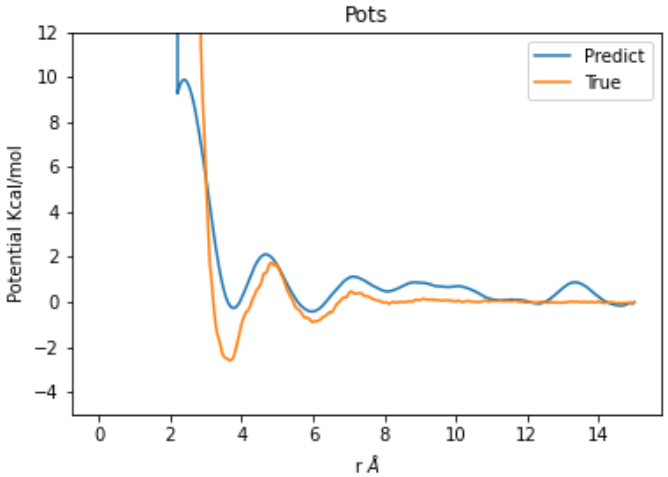

Fig. 4: Plot of True vs. Network Predicted Potentials

The (MSE) loss for this is ~3.36, which is considerably worse than the validation performance. This likely means that the generated trial potentials are not entirely suitable to describe the envelope/family of possible potentials.
It should also be noted that Mean Squared Error is not a perfect measure of accuracy for potentials, as the shape (slope) of the potentials is the actual physical measurement we care about, so it is very important for the predicted potential to have the correct features, even if the exact potential values are incorrect.

The slight disparity between the size of the forbidden areas is due to the RDFs including some areas which had very low (but non-zero) values. In future it is probably best to cut these regions out, as they likely belong to the forbidden area of the potential, but may be very briefly occupied (possibly due to initial system configuration).
Running simulations with these potentials gives the following RDFs, where Pred is the RDF obtained from using the network generated potential and True is the actual input RDF.

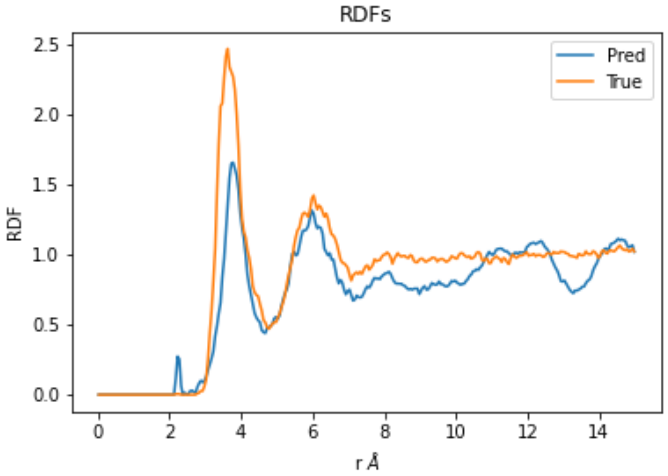

Fig. 5: Plot of True (Input) RDF vs RDF from using Predicted Potential (Pred)

For both the potentials and RDFs, the network has made a good approximation, which is likely sufficient for simulation in many cases.

Conclusion

In this project a method for generating CG potential for a given system was developed, but it is apparent that improvements can be made to improve accuracy of the calculated potentials. Currently, the accuracy of the produced potentials is sufficient for many simulations, but not for simulations that require higher degrees of accuracy.

Further Work

I believe that the primary area for improvements should be in the creation of the test potentials used to generate the training dataset. These trial potentials need to form an “envelope”, within which the true potential lies. It is important that the true potential is within the envelope, as the network is only capable of interpolating within the envelope, not extrapolating information outside of it.
It is also recommended to change the method for defining the forbidden region from the initial (input) RDF(s), instead of only taking the region where the RDF is 0, the region should also include the area where the RDF is suitably low (perhaps ~0.1). This potential in this region is also likely to be very high, but due to initial conditions the region may be very briefly occupied at the start of simulation.
The accuracy of the network when applied to systems with larger numbers of beads will also need to be investigated.

Github

The code, files used and worked example have all be uploaded to https://github.com/k-nolan-git/Summer_Project