MACHINE LEARNING RESULTS LOG SUMMER 2018 (AUGUST - PRESENT)

KIRK SWANSON

1. 8/6/2018

(1) Going back to tutorial section about implementing an example GCN. On Midway 2 gpu2, following the usual instructions.
(2) Briefly doing the PyTorch introduction: https://pytorch.org/tutorials/beginner/blitz/tensor_tutorial.ht

2. 8/8/2018

(1) On https://github.com/rusty1s/pytorch_geometric, looking at torch_geometric/data/dataset.py. We will go through this and understand line-by-line, because we want to create and load our own custom dataset.
(2) "import collections". From python documentation, collections "This module implements specialized container datatypes providing alternatives to Python's general purpose built-in containers, dict, list, set, and tuple. Counter, for example, counts the number of examples of each class in a list. "os.path". From doc, "This module implements some useful functions on pathnames. To read or write files see open(), and for accessing the filesystem see the os module."
(3) Now reading the Data Loading and Processing Tutorial on PyTorch: https://pytorch.org/tutorials/begi "torch.utils.data.Dataset is an abstract class representing a dataset. Your custom dataset should inherit Dataset and override the following methods: __len__ so that len(dataset) returns the size of the dataset, and __getitem__ to support the indexing such that dataset[i] can be used to get the ith sample." Transforms can often be written using __init__ method as well as __call__ method.

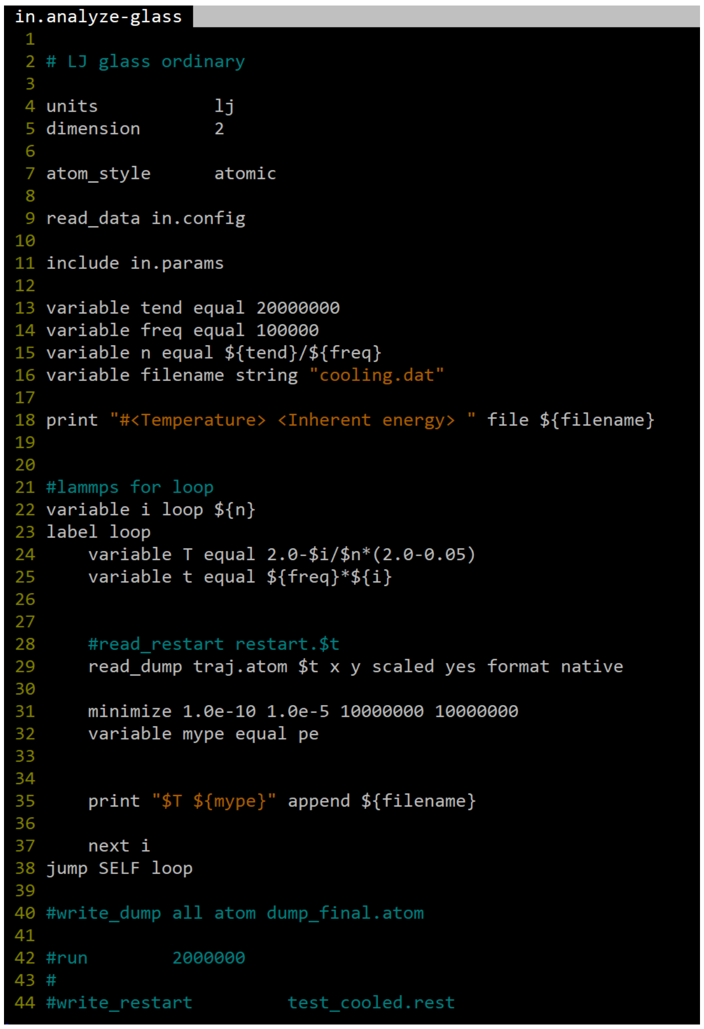
3. 8/20/2018

(1) The following describes issues related to the inherent structure energy, E_{IS}, computed in liquid-cooled simulations used for my proposed research project, "Deep Learning for Glassy Physics." The question of the origin of E_{IS} values came up during a talk I was giving to Professors de Pablo, Galli, and Skinner on 8/19/2018, with Rovana present. Professor Skinner asked what my computed E_{IS} values represented, which are on the order of -4.0 (Lennard-Jones units) for a binary mixture of 4,320 particles interacting according to a Lennard-Jones force. I suggested that these values represent the total energy of a minimized configuration, while he argued that they represent the energy per particle of a configuration. I was unable to resolve the issue on the spot and instead said that I would be happy to review the issue after the talk.

Date: October 3, 2018.

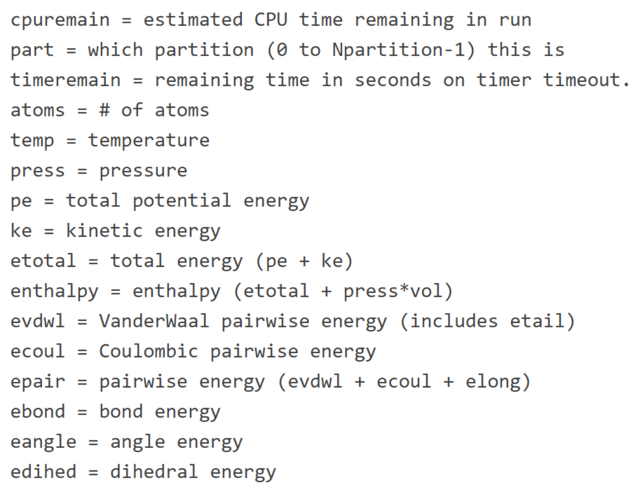
In order to resolve the issue, and to explain why I was suggesting that these E_{IS} values were total energy values, I first turned to the paper "Age and structure of a model vapour-deposited glass," by Reid et al. My liquid-cooled simulation parameters and methdology are derived directly from this paper, not only because it provides clear instructions for performing this type of computation, but also because I wanted to ensure similarity between my data and that of Reid et al. My machine learning project is attempting to answer questions raised in this paper, so it is important to produce simulations that have comparable characteristics. For their 2D liquid-cooled systems, Reid et al. measure inherent structure energy values also to be on the order of -4.0. The paper explains, "The inherent structure energy E_{IS} of a configuration, used to quantify its stability, is the potential energy of a configuration brought to its local energy minimum." This definition clearly suggests that E_{IS} is a total energy value, and no where else in the paper is the energy defined to be a per atom or per particle quantity.

The following shows the LAMMPS script that I used to compute inherent structure energy values from snapshots of a simulation:



```
in.analyze-glass

# LJ glass ordinary

units           lj
dimension       2

atom_style      atomic

read_data in.config

include in.params

variable tend equal 20000000
variable freq equal 100000
variable n equal ${tend}/${freq}
variable filename string "cooling.dat"

print "#<Temperature> <Inherent energy> " file ${filename}


#lammps for loop
variable i loop ${n}
label loop
    variable T equal 2.0-$i/$n*(2.0-0.05)
    variable t equal ${freq}*${i}


    #read_restart restart.$t
    read_dump traj.atom $t x y scaled yes format native

    minimize 1.0e-10 1.0e-5 10000000 10000000
    variable mype equal pe


    print "$T ${mype}" append ${filename}

    next i
jump SELF loop

#write_dump all atom dump_final.atom

#run        2000000
#
#write_restart         test_cooled.rest
```

As shown above, a trajectory is read into the system, and then configurations every 100,000 steps are analyzed. This analysis consists of three steps: perform an energy minimization (line 31), compute an energy (line 32), and print these values to an external file (line 35). The energy computation on line 32, "variable mype equal pe" is responsible for computing the inherent structure energy. The energy is taken from "pe", which is a variable pre-defined in LAMMPS and is described in official online documentation. The documentation clearly defines "pe" as a total energy value. The following is a screenshot from https://lammps.sandia.gov/doc/thermo_style.html:



```
cpuremain = estimated CPU time remaining in run
part = which partition (0 to Npartition-1) this is
timeremain = remaining time in seconds on timer timeout.
atoms = # of atoms
temp = temperature
press = pressure
pe = total potential energy
ke = kinetic energy
etotal = total energy (pe + ke)
enthalpy = enthalpy (etotal + press*vol)
evdwl = VanderWaal pairwise energy (includes etail)
ecoul = Coulombic pairwise energy
epair = pairwise energy (evdwl + ecoul + elong)
ebond = bond energy
eangle = angle energy
edihed = dihedral energy
```

Reid et al.'s definition of inherent structure energy, as well as the energy computation command in my code, clearly explain why I suggested that my E_{IS} values represent total potential energies rather than energies per atom. I am currently working to definitively confirm the origins of computed E_{IS} values to fully resolve this issue.

Much more importantly,however, the distinction between the total energy of a configuration and the average energy per atom of a configuration does not impact my project or my results. I am using this inherent structure energy value to compare the potential energy in different configurations that have the *same number of particles*. Therefore, making this comparison using total energy is equivalent to making the comparison using average energy per particle, and the type of comparison used has no impact on any aspect of the project or the results that I have so far collected.

4. 9/11/2018

(1) First thing we are going to do is to create an actual dataset folder on midway. Second thing will be to, step by step, build a custom dataset in PyTorch Geometric.
(2) On /home/swansonk1, directory called pytorch-geometric. Copying dataset there called data_coords_endpoints_restricted100, which has 100 files, 500 glass and 500 liquid, each containing the xy coordinates of 100 particles and particle type (I think most or all are type 1 in these subsets).

5. 9/12/2018

(1) Data objects in torch_geometric use the COO connectivity format for graphs. So, if we have the following adjacency matrix for a graph:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{5.1}$$

we represent this in condensed form as:

$$\begin{bmatrix} 0 & 1 & 1 & 2 \end{bmatrix} \\ \begin{bmatrix} 1 & 0 & 2 & 1 \end{bmatrix} \tag{5.2}$$

The first is row values, second column values, of the matrix indices where 1s are located. So, we have 1s at (0, 1), (1, 0), (1, 2), and (2, 1).

6. 9/16/2018

(1) In BASH terminal, type "source activate pytorch-env". In this environment, we have installed python 3.6, PyTorch for windows, and jupyter notebooks. Then, type "jupyter notebook –no-browser".
(2) Used "conda install pytorch-cpu -c pytorch" to install pytorch. Used "pip install torchvision", NOT pip3, to install torchvision.
(3) Had to pip install matplotlib. Perhaps next time when creating an environment, I should do the anaconda command that includes all relevant packages..
(4) On Midway 2 gpu2, module load Anaoncda3/5.0.1, module load cuda/8.0, module unload gcc, source activate geometric-env.

7. 9/18/2018

(1) Continuing tutorials. Doing Learning PyTorch with Examples. Saved in a jupyter notebook with this name on laptop /home/swansonkirk80 folder.

8. 9/19/2018

(1) Continuing to work on developing MPNN code in PyTorch. Working through creation of a Data class based on pytorch geometric repository.

9. 9/20/2018

(1) Looking on midway home, /.local/lib/python3.6/site-packages/torch_geometric/utils, the coalesce.py program.
(2) Realized that my pytorch-geometric version is essentially outdated, so now creating a new environment on midway. Did module load Anaconda3/5.0.1, module load cuda/8.0, module unload gcc. conda create -n pytorch-geometric-9-20-2018.
(3) On laptop, created environment called pytorch-geometric-9-20-2018-env, which has python 3.6, pytorch-cpu, and the latest pytorch-geometric installation. Used pip install for the latter. Does local.

10. 10/3/2018

(1) Made a small change to batch_PyTorch.py, which was missing a line batch[key].append(item).
(2) Wrote TestNN_PyTorch.py. Moving forward, we just need to work on developing the actual machine learning code!

E-mail address: `swansonk1@uchicago.edu`

INSTITUTE FOR MOLECULAR ENGINEERING, UNIVERSITY OF CHICAGO, 5640 S ELLIS AVE, CHICAGO, IL 60637