Journal Report 6
10/7/19-10/11/19
Sohom Paul
Computer Systems Research Lab
Period 1, White

Daily Log

Monday September 30

Debugging loading and preprocessing of data.
Learned actual content of script.

Tuesday October 1

Fixed redundant one-hot labeling.
Created hard-parameter sharing model.

Thursday October 3

Used multitask learning to simultaneously learn heat capacity and internal energy at 0 K.
Added tasks (dipole moment, HOMO) to above network and retrained.
Read papers on early stopping and Adam optimization.

Timeline

Date	Goal	Met
Sep 30	Build toy networks with multitask learning	Abandoned; stole code from Github instead.
Oct 7	Integrate multitask learning (hard parameter sharing) into first GCNs	Partial; need to modify to include separate dense layers.
Oct 14	Train network and compare results to tasks done without multitask learning.	Yes.
Oct 21	Build network with soft parameter sharing; modify loss accordingly.	
Oct 28	Finish building soft parameter sharing model. Compare to results with single-task learning and hard parameter sharing.	

Reflection

When I tried training my network on Monday, I was hit with some errors because the dimensions of certain numpy arrays not lining up. I knew that this must have been an error on my end, because the relevant lines had been stolen from a (working) GCN example from online. However, in searching for the bug, I ended up actually learning the code's logic, so it was probably ultimately beneficial that I messed up. For example, I hadn't realized that the representation of our molecules only considers the heavy-atom skeleton (probably to avoid needlessly introducing parameters for the hydrogen atoms, whose effect is largely inconsequential). Also, I learned that the EdgeConditionedConv that my partner was using was actually hiding learned neural convolutions, as opposed to the naive averaging of hidden states that I had assumed. In the future, we will need to compare the results of learned edge-conditioning to more traditional graph convolutions and see if a hand-tuned algorithm can improve accuracy and/or training time.

On Tuesday, I realized that the source of my bug was me running the same Jupyter cell - responsible for replacing my input with a one-hot encoding - twice. Once I realized that, the rest of class was finishing the code for hard parameter sharing and training the network. The results I got this week from multitask learning was much more accurate than last weeks. Included below (Figures 1 and 2) are the graphs of the predicted versus actual values for the (normalized and dimensionless) heat capacity and 0 K internal energy. Also, out of curiosity, I decided to compare (Figure 3) the loss versus epoch number for the single task training and multi task training for heat capacity.

We can see that, in the case of multitask learning for 2 tasks, the losses with and without multitask learning are comparable. However, some time in the future, I will have to do a more in-depth analysis to check correlation between each of our tasks. For example, when we perform multi task learning for heat capacity, 0 K internal energy, dipole moment, and HOMO energy, we get the follow losses:

Task	Loss
Heat Capacity	0.05138
Internal Energy	0.04374
Dipole Moment	0.52670
HOMO	0.28500

Notice that the dipole moment and HOMO losses are huge compared to the heat capacity and internal energy losses. However, this might be indicative of the fact that I did not giving enough training time to learn dipole moment and HOMO. The losses for heat capacity and internal energy were higher in this 4-task network when compared to the 2-task network, although this might be due to the fact that I decreased the number of training epochs and examples in an effort to be spending less time just sitting in front of the computer, waiting for the network to train. (I'm starting to run out of relevant papers to read.)

For the future, I decided to extend implementing soft-parameter sharing from a one week goal to a two week goal, considering that it took me longer than expected to do the much simpler task of implementing hard parameter sharing. Also, as a more nebulous goal, as the year progresses, we will be training larger and more intricate networks, so it will be necessary for me to gain access to GPU. (The one on my laptop won't do because I'm using Windows Subsystem for Linux, so the version of Tensorflow I have installed won't interface correctly with my hardware drivers. In other words, Windows is bad.) We will need to make sure that all the libraries we need are properly installed on TJ machines - but that's a problem for next-to-next month, most likely.

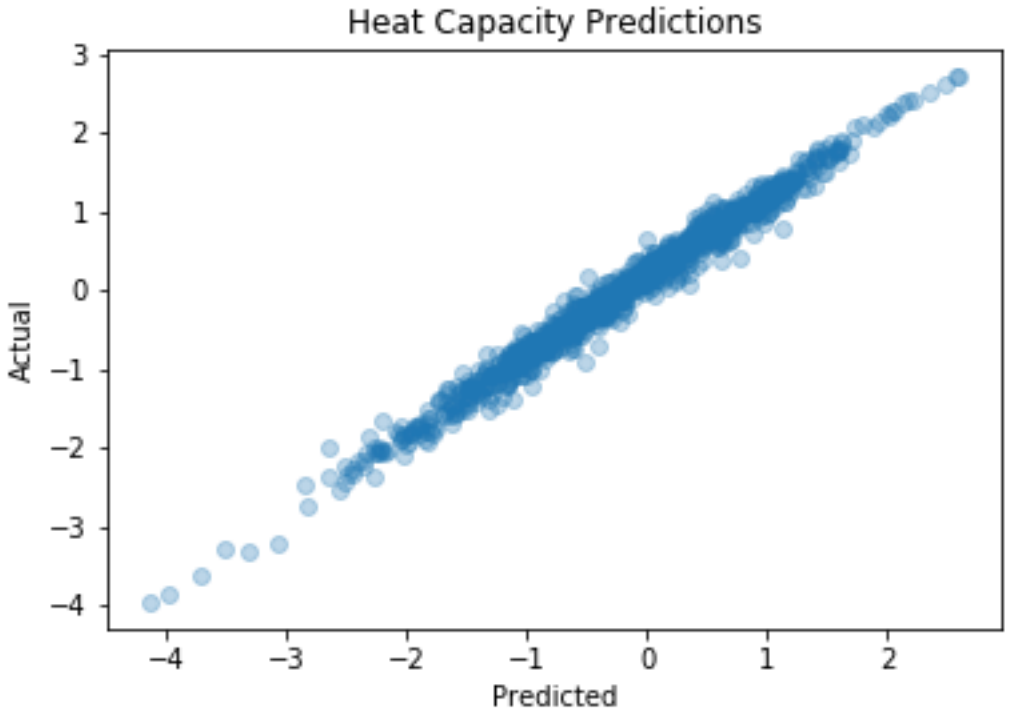

Figure 1: Accuracy of prediction of (normalized) heat capacity for two-task learning.

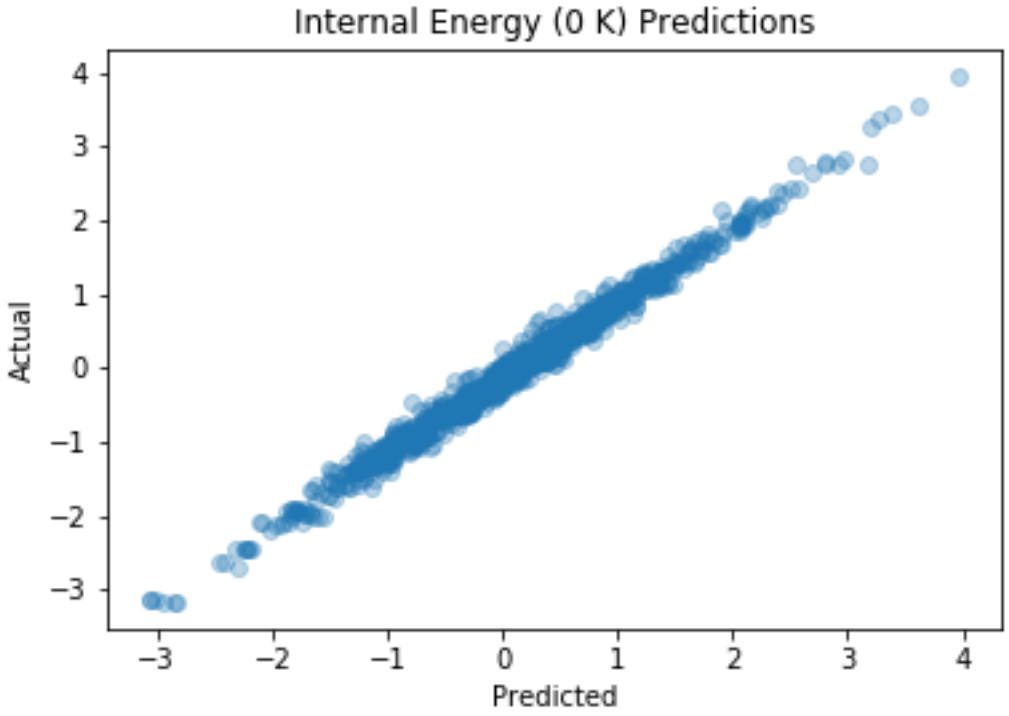

Figure 2: Accuracy of prediction of (normalized) internal energy for two-task learning.

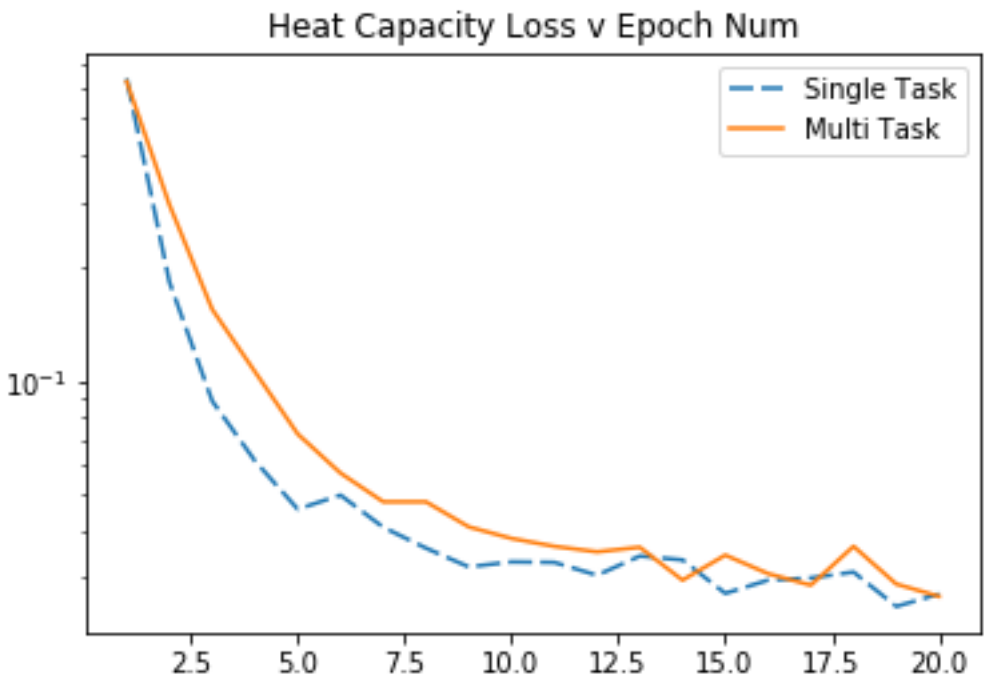

Figure 3: Comparison of single task and multitask learning with respect to loss of the heat capacity task with respect to epoch number.