Journal Report 9
11/11/19-11/15/19
Sohom Paul
Computer Systems Research Lab
Period 1, White

Daily Log

Monday November 11

Started splitting model into encoder and classifier to allow for cross-task transfer coefficient calculations.
Encountered bug when trying to chain models with Keras' functional API.

Tuesday November 12

Continued trying to split model into encoder and classifier to allow for cross-task transfer coefficient calculations.
Solved bug with chaining models.
Created bug with lack of differentiability.

Thursday November 14

Reorganized code and pushed to Github repo.
Verified functioning of Psi4 scripts.
Continued debugging model split.

Timeline

Date	Goal	Met
Nov 4	Find algorithm to perform task clustering. Perform task clustering.	Yes
Nov 11	Implement neural clustering (CMTL) and compare with non-clustered results	Partial; I've yet to compute the full 19x19 table of coefficients.
Nov 18	Finish performing neural clustering. Start implementing deep relationship networks.	Postponed, due to errors in last week's work.
Nov 25	Finish performing neural clustering. Generate 19x19 table of task relatedness. Start building demonstration program with DFT and single-task results.	
Dec 2	Continue building demonstration with preliminary multitask results.	
Winter Goal	Have a program capable of taking an input molecule and predicting the relevant molecular properties, using a choice of density functional theory or multitask-learned networks. The demo will let the use see firsthand the relative accuracies and speeds of the methods.	

Reflection

As I've indicated my journal for the past week, while I had thought that I had completed my code for neural task clustering, what I had really done was simply measure the L2 distance between the results of each task, which is no better than doing clustering based on covariances. Thus, I had modified my goal for this week to be to fix my error and build a legitimate neural task clustering algorithm. I have failed in this task.

The idea behind neural task clustering is comparing the similarities in the learned internal representation of each convolutional neural network. Thus, one needs to split the network into a encoder and classifier (where I'm using the word "classifier" loosely here, considering that the final output is real number rather than a category label) and calculate the losses for each task when the encoders are swapped out with the encoders from other tasks. I have spent much of the past few class periods working on this but I've yet to produce working results.

When I first tried making the split between encoder and classifier, I has having an error thrown by some obscure part of the Keras source code. I did not think to write down the error, but it had something to do with the dimensions of certain tensor not matching up in an np.einsum call. I was able to get rid of this error by following a GAN tutorial and structuring my code in the same way. However, for Tuesday and Thursday classes, I've been plagued by an error about the differentiability of my model's functions. I did not make any explicit calls to nondifferentiable functions and I'm at a loss as to what's causing it. (Granted, ReLU is technically not differentiable,

but I'm certain that the gradient has been hard-coded somewhere because it works in the models I've trained before.)

I'm going to see for the next couple of classes if I can catch and fix the bug. If I can't, then, in interest of spending my time effectively, I'm going to pivot into building the demonstration for my Winter Break goal. I spent some time last class working on making sure that my code from earlier in the year for running density functional theory calculations still works. (Amusingly, it failed initially because I didn't have all the packages I'm using in the same conda environment.) Once I have the demonstration working for the code I wrote earlier in the year, then I can return to my original goals of neural task clustering and building deep relationship networks.