**From:** **Tiago Azevedo** tiago.azevedo@cst.cam.ac.uk
**Subject:** Developments for the part II project
**Date:** 23 August 2019 at 21:39
**To:** K. Stankeviciute ks830@cam.ac.uk
**Cc:** andreeadeac22@gmail.com, Dr Pietro Lio pl219@cam.ac.uk

Hi Kamile!

How are you? As promised I thought about tasks to include in your possible project proposal. At this point I think it depends on your interests. because we really have many possibilities (I've written a bigger email than I was expecting). I'm putting here some bullet points with my thoughts, let me know what you think that could be more interesting. Maybe in the meantime you also got more ideas of your own, and you might not like some parts I've written.

If we go into the "reproduce paper and develop some extensions" path, you can make that slightly more interesting:
- You reproduce the paper but using some of the new geometric DL tools like pytorch geometric or DGL instead of tensorflow as in the paper. This would be good because (1) you can argue using these tools make the code easier to be used by peers (usability) because you'd be using a specific tool for a specific type of structured data, also making it easier to be extended thus having proper software engineering methods for future extension, (2) you learn new tools (I guess), and (3) you can argue this is good for a part II project as you have quite a lot of code implementation to do.

The following points for extensions/tasks are probably relatively easy to implement if you reproduce the paper, and could be an easy way to have more content in your dissertation, at the same time as you have the freedom to focus and learn things that you never explored before:

- You try a different dataset (thus probably a different problem), to try to highlight how generic the approach is in terms of the problem being tackled. This might be tricky because you may end up having to learn boring scripts to generate similar features on non-preprocessed data. Which could be related to the next point.

- Explore missing data behaviour. What about making it a semi-supervised learning but a bit different from what is described in the paper. For example, if you remove 1%/5%/10% of data, how robust is the propagation of GCN to other nodes with missing data (for example randomly initiated features). In other words, how can we correctly predict one person's outcome by just considering the phenotypical relations around him/her? Maybe for small percentages we could still get some nice metrics (obviously not sure).

- Or try other types of data in ADNI/ABIDE. I've checked these datasets and looks like ADNI has gene expression data available, as well as some sort of methylation data. Might be worth to check if this comes in a format which you could actually use. From ABIDE it doesn't look like we have any genetic data available.

- You could try some other dimensionality reduction techniques. For example, they used a simple autoencoder, but variational autoencoders and others already proved to be much powerful. This would be quite an easy way to have extra tasks and with a certain possibility of improving your final metrics.

- You could try some other approaches other than GCN. I mean, there are sooo many (https://pytorch-geometric.readthedocs.io/en/latest/modules/nn.html).

Regarding the limitations they mention in the discussion section of the paper:
- They mention changing the problem to a multi-class classification or even a regression. They don't specify how they would get the labels, but I'm guessing in this case they would be predicting disease severity (for example using some assessment score like MMSE). They seem to have a few available in ADNI, in this case you might need to search which ones are more interesting/reliable.

- The suggestion of using the Cayley polynomials, although I don't know them, could be a straightforward way to go an extra task. It looks like it makes sense to try them instead of the ones they use to generate the edges.

- Regarding the point of making it a dynamic graph, I'd say this is definitely interesting, although a different type of problem. The paper they cite as an example of solution (Monti's paper) looks like is implemented in pytorch_geometric. But you also have other methods if you want to go with this path.

- The graph construction techniques they mention are also a possibility. One particular that I think is more interesting is using more than 1 feature per edge, instead of just an edge either existing or not.

Cheers,
Tiago.