Project proposal notes

Table of contents:

Inspiration for the project

The papers and the code (described below) describe a semi-supervised graph convolutional network (GCN) to predict

- whether the patient suffers from Autism Spectrum Disorder (ASD) and
- whether the patient suffering from Mild Cognitive Impairment (MCI) will develop Alzheimer's disease (AD).

The graph is used to exploit both

- *the individual features* of patients (stored as vertex features in the graph) and the
- *similarity between patients* (represented as possibly weighted edges in the graph).

This is thought to be better than using just graphs (which ignore individual features of patients) or just non-graph classifiers (which would not allow for patients to share the data and infer the diagnosis from patient's neighbourhood, especially when imaging data is not available for all patients). The graph also allows to exploit *multimodality* of data—the approach uses both imaging (fMRI, MRI, brain volume, longitudinal brain scans etc.) and non-imaging (gender, age, acquisition site (determining the imaging data collection protocol), possibly genetic) data.

Papers

The main papers on which the project would be based are:

Parisot, S., Ktena, S. I., Ferrante, E., Lee, M., Moreno, R. G., Glocker, B., & Rueckert, D. (2017).

Spectral Graph Convolutions for Population-based Disease Prediction.
MICCAI 2017.

and

*Parisot, S., *Ktena, S. I., Ferrante, E., Lee, M., Moreno, R. G., Glocker, B., & Rueckert, D. (2017).
Disease Prediction using Graph Convolutional Networks: Application to Autism Spectrum Disorder and Alzheimer's Disease.
Medical Image Analysis, 2018.

The graph convolutional network used in the implementation is first introduced in

Thomas N. Kipf, Max Welling (2017).
Semi-Supervised Classification with Graph Convolutional Networks.
ICLR, 2017

Codebase

The code used for the papers above (**applied to ABIDE database but not for ADNI**) is publicly available at https://github.com/parisots/population-gcn.

The implementation for ADNI should be similar, but would need:

- coding the data parser and processor from scratch;
- adapting the model to different features and different similarity metrics.

which could be a fair amount of work.

The GCN is based on https://github.com/tkipf/gcn (and further customised for the papers). It uses TensorFlow which does not support deep learning on graph structured data as well as some other libraries/frameworks like `torch_geometric` (PyG) or Deep Graph Library (DGL) would, and so is less extensible/applicable to other problems and datasets.

Datasets

The project will primarily only use ADNI for Alzheimer's disease as the benchmark, but some ideas could incorporate such datasets as ABIDE (ASD) and PPMI (Parkinson's disease) as alternative datasets/benchmarks:

- ADNI (Alzheimer's Disease Neuroimaging Initiative): http://adni.loni.usc.edu [*Access requested and retrieved*]
- ABIDE (Autism Brain Imaging Data Exchange): http://fcon_1000.projects.nitrc.org/indi/abide/
- PPMI (Parkinson's Progression Markers Initiative): https://www.ppmi-info.org/

Reimplement + extend

This seems to be a common approach to Part II projects and I like that it gives some direction to what should be achievable (i.e. at least reproducing the results).

This approach also currently has more ideas for the project.

Addressing some problems with the current implementation could lead to new state-of-the-art results while making the API more accessible and extensible.

The sections below list the things that could be considered as part of the project. They also list the potential goals that could be set in the project and used as success criteria, such as the following:

Goal: at least 80.0% accuracy for the ADNI dataset in classifying MCI to AD progression in patients.

Using another geometric deep learning library

The original implementation was TensorFlow with a custom implementation of a GCN. This indicates the lack of common TensorFlow APIs for graph processing. Some alternatives (both in Python) that could improve the usability of the code and make it more flexible are:

- PyTorch Geometric (PyG)
- Deep Graph Library (DGL)

These libraries are also nice because they could be easily used to *customise the graph neural network* and potentially explore other methods other than graph convolution. For example, the list of *geometric deep learning methods* in PyG is available at https://github.com/rusty1s/pytorch_geometric/blob/master/README.md (the graph convolution method used in the papers is available as `GCNConv` package).

Goal: find the convolution method that could give better performance (measured in accuracy, speed or some other metrics described below).

Evaluating the robustness of GCNs

The method proposed in the original papers relies on semi-supervised classification, therefore the decision for some patient could depend on the information of the patient's neighbourhood. This is especially useful when the patient in question has *incomplete data* (e.g. no imaging data) but the neighbouring nodes have more information as well as the diagnosis. In this way the graph structure helps to spread some patterns occurring in the population between the individuals.

Question to explore: how robust is the graph performance to missing or incorrect data?

Would accuracy stay the same if 5% of the labels/features/edges were missing? What about 10%? Would other geometric deep learning (GDL) libraries perform better in this metric?

This is important if we were to add new patients to the population (with incomplete information) and would want to find out their diagnosis. This would also indicate how much information do we need to have about the patient to correctly determine if they are healthy or suffering, or what proportion of *incorrect labels* can the algorithm tolerate. It could also indicate the *ability to generalise* to other datasets.

Goal: measure robustness of the baseline implementation and find a more robust algorithm. [Robustness to incorrect labels and/or missing data]. I like that this question is more on the computer science than pure software engineering side. In any case it also serves as a *good evaluation criterion* to compare the baseline model to any extensions—proving the *new model does not overfit* the data.

Incorporating more types of data

ADNI database contains various clinical, genetic, MRI image, PET image and biospecimen data (see full description at http://adni.loni.usc.edu/data-samples/data-types/).

However, the original papers only use a limited set of features to construct the graph. In particular,

- Feature vectors (for vertices) are derived purely from the *volumes of segmented brain structures*, which have proved to be highly effective in classifying stable vs. progressive MCI. *Feature selection was not used*, because of "much smaller and tractable feature vector size".
- Graph edges are based on *sex and gender* information because this information highly affects feature vector values—i.e. it was aimed to connect the nodes that simply share age and gender to connect related brain volumes.

There is potential for

- exploring if any other features might be useful apart from combinations explored in the papers (*sex, age, acquisition site*)
- *learning* the edge weights instead of being binary (edge exists if sex/age of the adjacent vertices matches)

Goal: explore more similarity metrics and feature combinations to improve performance.

Other ideas

Dimensionality reduction

The 2018 paper mentions that this is *irrelevant to ADNI database* because the imaging data features are tractable without this.

On the other hand, **if new features were used** dimensionality reduction could be one approach to manage the new data. Some methods that could be considered

- PCA
- autoencoders and VAEs

Using Cayley polynomials instead of Chebyshev polynomials

These polynomials are used to efficiently compute the graph convolutions. The 2018 paper uses Chebyshev polynomials but mentions that the recently introduced Cayley polynomials [1] could improve the results.

[1]: Levie, R., Monti, F., Bresson, X., Bronstein, M.M., 2017. Cayleynets: Graph convolutional neural networks with complex rational spectral filters. arXiv preprint arXiv:1705.07664.

Solving a related problem

This section looks at a broader set of problems that go outside of improving the results of this particular paper.

Application to another problem: dynamic graphs

The 2018 paper mentions that the limitation of GCNs/Chebyshev polynomials is that the graph needs to have a known structure. This means that once the model is trained the graph cannot be altered (and if it does the model needs to be retrained).

It could be an interesting problem to support dynamic graphs (*adding*) a patient with unknown diagnosis, but it has been suggested that this change would not really add much value—the ultimate goal is to *understand the patterns* that the suffering patients share rather than simply classify a patient, a doctor can do that themselves :)

Application to another dataset

GCNs could be used to classify the patients of another disease, for example using the PPMI dataset for Parkinson's disease.

In this case success criteria could be tricky because I can't know the accuracy that is achievable using the GCN method in advance.