

**Introduction**

This work has been published in MedIA 2020.

netNorm is a framework used to estimate centered and representative multi-view brain connectional templates.

Considering a population of brain networks where each subject is represented by a tensor presenting a set of brain views (multi-view data), netNorm first selects the most representative cross-view feature vector in the population for each pairwise connection. Therefore, it creates a population representative tensor that only includes the most centered feature vectors. Then, netNorm non-linearly fuses the frontal views of the estimated representative population tensor into a single network using SNF technique depicting the final brain connectional template. We have evaluated our method on morphological brain networks for healthy and autistic subjects. But we note that netNorm can be used on any type of multi-view datasets.

Detailed introduction can be found in the original paper.

In this repository, we release the code to train and test netNorm on multi view networks. We have also added the estimated CBTs (connectional brain templates) for the data used on the paper for ASD population (subjects with autism spectrum disorder) and NC (normal controls) for both left and right hemispheres (RH and LH).

**Citation**

@article{dhifallah2020estimation,

title={Estimation of connectional brain templates using selective multi-view network normalization},

author={Dhifallah, Salma and Rekik, Islem and Alzheimer's Disease Neuroimaging Initiative and others},

journal={Medical Image Analysis},

volume={59},

pages={101567},

year={2020},

publisher={Elsevier}

}

**Installation**

This code has been tested with Matlab 2018a on Windows 7.

**Usage**

**Data Preparation**

We used random data sets for test generated using the function “simulateData.m”. The number of views, the number of subjects and ROIs (must be >20) are fixed by the operator. In order to test netNorm on other datasets, upload the data as a cell variable of size . Each cell represents a view tensor of size where is the number of ROIs and is the number of subjects. Every stacked matrix in the tensor represents a brain network for a subject. Please make sure to include the same order across all views.

**Train and test netNorm**

Train and test netNorm method using:

netNorm.m

**Acknowledgment**

We used the following code:

SNF code from <http://compbio.cs.toronto.edu/SNF/SNF/Software.html>.

**Related references**

Similarity Network Fusion (SNF): Wang, B., Mezlini, A.M., Demir, F., Fiume, M., Tu, Z., Brudno, M., HaibeKains, B., Goldenberg, A., 2014. Similarity network fusion for aggregating data types on a genomic scale.

[<http://www.cogsci.ucsd.edu/media/publications/nmeth.2810.pdf>] (2014) [<https://github.com/maxconway/SNFtool>].

**License**

Our code is released under MIT License (see LICENSE file for details).

# NAGFS (Network Atlas-Guided Feature Selection)

NAG-FS (Network Atlas-Guided Feature Selection) for a fast and accurate graph data classification code, created by Islem Mhiri. Please contact islemmhiri1993@gmail.com for inquiries. Thanks.

While typical feature selection (FS) methods aim to identify the most discriminative features in the original feature space for the target classification task, feature extraction (FE) methods cannot track the original features as they extract new discriminative features via projection. Hence, FS methods are more convenient for clinical applications for biomarker discovery. However, existing FS methods are generally challenged by space, time, scalability, and reproducibility. To address these issues, we design a simple but effective feature selection method, which identifies the most discriminative features by comparing healthy and disordered \*brain network atlases to learn\*.

![NAGFS pipeline](http://basira-lab.com/nagfs\_0/)

# Detailed proposed NAGFS pipeline

This work has been published in the Journal of Medical Image Analysis 2020. \*\*Network Atlas-Guided Feature Selection (NAG-FS)\*\* is a network atlas-based connectomic feature selection method for a fast and accurate classification. Our learning-based framework comprises three key steps. (1) Estimation of a centered and representative network atlas, (2) Discriminative connectional biomarker identification, (3) Disease classification. Experimental results and comparisons with the state-of-the-art methods demonstrate that NAG-FS can achieve the best results in terms of classification accuracy and overall computational time. We evaluated our proposed framework from ABIDE preprocessed dataset (http://preprocessed-connectomes-project.org/abide/).

More details can be found at: https://www.sciencedirect.com/science/article/pii/S1361841519301367 or https://www.researchgate.net/publication/337092350\_Joint\_Functional\_Brain\_Network\_Atlas\_Estimation\_and\_Feature\_Selection\_for\_Neurological\_Disorder\_Diagnosis\_With\_Application\_to\_Autism

![NAGFS pipeline](http://basira-lab.com/nagfs\_1/)

![NAGFS pipeline](http://basira-lab.com/nagfs\_2/)

# Folders to unzip before running the demo

Make sure you unzip the following files before running the demo:

1. circularGraph.zip <br/>

2. libsvm-3.23.zip <br/>

3. snnf.zip <br/>

4. src.zip <br/>

# Demo

The code has been tested with MATLAB 2018a on Windows 10. GPU is not needed to run the code.

In this repository, we release the NAGFS source code trained and tested on a simulated heterogeneous graph data from 2 Gaussian distributions as shown below:

![NAGFS pipeline](http://basira-lab.com/nagfs\_3/)

\*\*Data preparation\*\*

We simulated random graph dataset from two Gaussian distributions using the function simulateData.m. The number of graphs in class 1, the number graphs in class 2, and the number of nodes (must be >20) are manually inputted by the user when starting the demo.

To train and evaluate NAGFS code on other datasets, you need to provide:

• A tensor of size ((n-1) × m × m) stacking the symmetric matrices of the training subjects. n denotes the total number of subjects and m denotes the number of nodes.<br/>

• A vector of size (n-1) stacking the training labels.<br/>

• A number of selected features Nf.<br/>

• A boolean variables ‘displayResults’ ∈ [0, 1].<br/>

If displayResults = 1 ==> display (Atlas of group 1, Atlas of group 2, top features matrix and the circular graph \*\*at each cross-validation run\*\*). This is cool but it might slow down the demo a bit.<br/>

If displayResults = 0 ==> no display except for the average results across all cross-validation runs.<br/>

The NAG-FS outputs are:

• A matrix of size (m × m) storing the network atlas of group 1. <br/>

• A matrix of size (m × m) storing the network atlas of group 2. <br/>

• A vector of size (Nf × 1) stacking the indices of the top discriminative features. <br/>

\*\*Train and test NAG-FS\*\*

To evaluate our framework, we used leave-one-out cross validation strategy.

To try our code, you can use: run\_demo.m

# Example Results

If you set the number of samples (i.e., graphs) from class 1 to 30, from class 2 to 30, and the size of each graph to 60 (nodes), you will get the following outputs when running the demo:

![NAGFS pipeline](http://basira-lab.com/nagfs\_4/)

# Acknowledgement

We used the following codes from others as follows:

SIMLR code from https://github.com/BatzoglouLabSU/SIMLR/tree/SIMLR/MATLAB.

SNF code from http://compbio.cs.toronto.edu/SNF/SNF/Software.html.

CircularGraph code from https://www.github.com/paul-kassebaummathworks/circularGraph.

# Related references

Similarity Network Fusion (SNF): Wang, B., Mezlini, A.M., Demir, F., Fiume, M., Tu, Z., Brudno, M., HaibeKains, B., Goldenberg, A., 2014. Similarity network fusion for aggregating data types on a genomic scale. [http://www.cogsci.ucsd.edu/media/publications/nmeth.2810.pdf] (2014) [https://github.com/maxconway/SNFtool].

Single‐cell Interpretation via Multi‐kernel LeaRning (SIMLR): Wang, B., Ramazzotti, D., De Sano, L., Zhu, J., Pierson, E., Batzoglou, S.: SIMLR: a tool for large-scale single-cell analysis by multi-kernel learning. [https://www.biorxiv.org/content/10.1101/052225v3] (2017) [https://github.com/bowang87/SIMLR\_PY].

Paul Kassebaum (2019). circularGraph (https://www.github.com/paul-kassebaum-mathworks/circularGraph), GitHub. Retrieved December 26, 2019

# Please cite the following paper when using NAG-FS:

@article{mhiri2019joint,

title={Joint Functional Brain Network Atlas Estimation and Feature Selection for Neurological Disorder Diagnosis With Application to Autism},<br/>

author={Mhiri, Islem and Rekik, Islem},<br/>

journal={Medical Image Analysis},<br/>

volume={60},<br/>

pages={101596},<br/>

year={2020},<br/>

publisher={Elsevier}<br/>

}<br/>

Paper link on ResearchGate:

https://www.researchgate.net/publication/337092350\_Joint\_Functional\_Brain\_Network\_Atlas\_Estimation\_and\_Feature\_Selection\_for\_Neurological\_Disorder\_Diagnosis\_With\_Application\_to\_Autism

# License

Our code is released under MIT License (see LICENSE file for details).

# Contributing

We always welcome contributions to help improve NAG-FS and evaluate our framework on other types of graph data. If you would like to contribute, please contact islemmhiri1993@gmail.com. Many thanks.