



Institute of Information and Communication Technologies,
Electronics and Applied Mathematics

Machine Learning On Complex Networks: Dynamical Fingerprints, Embeddings and Feature Engineering

Leonardo Gutiérrez Gómez



Thesis submitted in partial fulfillment of the requirements for the degree of
Docteur en sciences de l'ingénieur

Thesis committee:

Advisor	Prof. Jean-Charles Delvenne	<i>Université catholique de Louvain</i>
Jury	Prof. Michel Verleysen	<i>Université catholique de Louvain</i>
	Prof. Marco Saerens	<i>Université catholique de Louvain</i>
	Prof. Renaud Lambiotte	<i>University of Oxford</i>
	Dr. Rémy Cazabet	<i>Université de Lyon 1</i>
	Dr. Francesco Bonchi	<i>ISI Foundation</i>
Chairman	Prof. Roland Keunings	<i>Université catholique de Louvain</i>

Remerciements

Le début de mon projet de doctorat a commencé avec “la Recherche”.

En effet, le magazine français *La Recherche* publiait, il y a quatre ans, dans son édition 482, un extraordinaire article scientifique concernant la privacité et l’analyse des données téléphoniques, témoigné notamment par des chercheurs de l’UCLouvain: Adeline Decuyper et le Prof. Vincent Blondel. C’est avec eux que je voulais commencer mes remerciements, puisque grâce à eux j’ai pu découvrir la pointe de l’iceberg de la recherche scientifique à l’UCLouvain, et ils m’ont donné l’envie de faire ma recherche en Belgique.

Joindre l’ICTEAM ¹ comme étudiant doctorant a été une des aventures les plus formidables que j’ai jamais eu. Néanmoins, ça n’aurait pas pu avoir été aussi génial sans les gens extraordinaires que j’ai rencontré et avec qui j’ai partagé des moments mémorables dans le cadre personnel et professionnel.

Je veux tout d’abord remercier l’équipe du secrétariat, Marie-Christine, Nathalie et Pascale pour avoir été super efficaces dans leur travail et avoir rendu ma vie quotidienne plus facile. En outre, le support technique fourni par Etienne et François n’aurait pas pu être meilleur, un grand merci.

Je remercie énormément toutes les personnes qui depuis le début de ma thèse jusqu’à la fin m’ont conseillé et encouragés pendant cette aventure. Notamment, les membres du *networks science group*, Leto Peel, Mauro Facin, Michael Fanuel, Michael Shaub, Alexey Medvedev, Jerome Kunegis, Martin Gueuning, Mridul Seth, un grand merci pour les bons repas, les conférences académiques, les groupes meetings et les discussions intéressantes que nous avons partagés ensemble. Parmi eux, je suis spécialement reconnaissant envers mes office mates et coauteurs, Benjamin Chiêm and Alexandre Bovet pour le bon travail achevé ensemble, c’était extraordinaire. De plus, un grand merci à mon coauteur Jakub Vohryzek et mes collaborateurs du CHUV ², le Pro-

¹Institute for Information and Communication Technologies, Electronics and Applied Mathematics

²Centre hospitalier universitaire vaudois

fesseur Patric Hagmann et ses collègues du centre psychiatrique de l'université de Lausanne.

Faire de la recherche demande de l'expertise, de la sagesse et surtout de l'expérience. C'est pour cela que je veux profondément remercier mon promoteur Jean-Charles Delvenne pour son encadrement. Il a toujours été présent pour m'écouter, me conseiller et m'encourager. La sérénité, la rigueur et l'élégance scientifique sont des valeurs que j'apprécie chez lui. Merci de m'avoir guidé dans l'aboutissement de ma recherche. Merci pour ces moments de travail intenses avant les deadlines et le cours de rédaction scientifique en anglais :). Ça a été un véritable privilège de travailler avec toi.

Je voudrais exprimer mes remerciements aux membres de mon comité de accompagnement et membres du jury, les Professeurs Michel Verleysen, Renaud Lambiotte et Marco Saerens pour leurs conseils pendant les différentes étapes de mon parcours. Je remercie également Rémy Cazabet et Francesco Bonchi pour avoir accepté de faire partie de mon jury de thèse. Merci au Prof. Roland Keunings de présider le jury.

Je remercie beaucoup les membres du personnel du bâtiment Euler avec lesquels j'ai partagé des bons gâteaux, BBQs et ICTEAM days amusants. Merci à Emilie R., Estelle M., François G., Aaron R., Benjamin V., Carlos R., Philippe D., Adrien T., Astrid L., Costanza C., Shuyu D., Pierre-Antoine B, ainsi qu'aux Professeurs Julien H., Raphael J., Pierre-Antoine A., Paul Van D., François G. et Laurent J. Merci aussi aux chercheurs du Maxwell, Cyril D.B., Dounia M., Dimitri De S., et John Lee et son chercheur visiteur, Daniel.

En dehors du travail, j'ai pu compter sur le soutien de nombreux amis. Je remercie Cecilia G., Bea M., Mike J., Barbara C., John V.N., Quentin D., Olivia L., Nacho et Laura pour les moments mémorables qu'on a partagés ensemble. Je tiens également à remercier mes amis de vie, mon pote Nico, Marco Polo, Justine, Valentina et Paolita, merci pour leurs soutiens. Également, je remercie les encouragements reçus de la part de Clem, Loukoum, Laura, Simon, Julien, Sophie, Amélie, Sarah, Alex, et Louise.

A ma compagne, ma complice, mon amoureuse Laëtitia, je te remercie profondément pour tout le soutien et ton amour inconditionnel tout au long de cette aventure. Merci beaucoup Geneviève et Jean Luc pour vos encouragements.

Mes remerciements finaux et les plus importants sont pour ma famille. Merci pour leur grand amour qui traverse l'Atlantique et leur soutien au quotidien. Gracias familia por su apoyo, los amo.

Complex networks emerge as a natural framework to describe real-life phenomena involving a group of entities and their interactions, i.e., a social network. Furthermore, other problems involve a collection of networks, such as multi-layer, temporal, or brain networks (connectomes). With the increasing availability of graph-shaped data and associated meta-data, i.e., node or edge attributes, machine learning (ML) techniques on networks have risen in popularity. However, there are still several challenging issues to be addressed concerning feature engineering, graph representation, visualization, and general graph mining applications.

This thesis contributes to bridging the gap between ML and complex networks. We investigate dynamics on networks such as discrete and continuous-time random walks, in two directions. First, we develop a multi-scale anomaly detection algorithm on attributed networks. We exploit the link between graph signal processing and the Markov stability framework used in community detection, for spotting anomalous nodes in multiple contexts of the network. Second, we introduce a generalization of assortativity on networks operating on scalar and categorical node attributes. These coefficients turn out to be useful descriptors to build network fingerprints, so that we can perform supervised graph classification, i.e., predicting the toxicity of molecules or classifying Reddit discussion threads.

Besides, we develop an unsupervised approach to learn graph embeddings from a collection of networks. By learning a non-linear mapping from input graphs to a lower-dimensional feature space, we obtain useful graphs representations used in graph visualization, clustering, and classification, e.g., predicting people's gender based on their structural connectome. Finally, we propose a principled approach to identify stable biomarkers for schizophrenia diagnosis in the human connectome. From an ML perspective, the problem is stated as an embedded feature selection scheme for graph classification. By doing so, we identify the affected-core in schizophrenia, highlighting increasing performance when structural and functional neural features are combined.

1	Introduction	17
1.1	Context and motivation	17
1.2	Contributions and structure of the thesis	19
1.3	Publications	22
2	Background and prerequisites	25
2.1	Introduction to complex networks	25
2.1.1	Graph theory	25
2.1.2	The graph Laplacian	27
2.1.3	Assortativity mixing	28
2.1.4	Community detection on networks	29
2.1.5	Random walks in networks	30
2.1.6	Introduction to the Markov stability framework	32
2.2	Introduction to machine learning	36
2.2.1	Feature selection and extraction	37
2.2.2	Supervised learning	38
2.2.3	Unsupervised learning	43
3	Anomaly detection on attributed networks	47
3.1	Introduction	48
3.2	Filtering graph signals and clustering attributed networks	51
3.2.1	Graph signals on attributed networks	51
3.2.2	Filtering of graph signals	52
3.2.3	Clustering attributed networks	53
3.2.4	Anomaly detection	54
3.2.5	Context for anomalous nodes	56
3.3	Experiments	57
3.3.1	Synthetic dataset	57
3.3.2	Real life datasets	58

3.3.3	Experimental settings	59
3.3.4	Results	61
3.3.5	Computational Issues	62
3.4	Multi-scale anomaly detection: a case of study	63
3.5	Related methods	65
3.6	Conclusions and perspectives	66
4	Multi-hop assortativities for networks classification	69
4.1	Introduction	70
4.2	Random walks and multi-hop assortativities	73
4.3	Features of network dynamics	76
4.4	Applications in Graph Classification: experiments and results	80
4.4.1	Datasets	80
4.4.2	Experimental setup	81
4.4.3	Results	83
4.4.4	Computational Cost	87
4.5	Related methods	88
4.6	Conclusions and perspectives	90
5	Network embeddings with node identity awareness	93
5.1	Introduction	94
5.2	Methods	97
5.2.1	Graph distances	97
5.2.2	Embedding distances	98
5.3	Experiments and Results	101
5.3.1	Synthetic datasets	102
5.3.2	Applications in graph visualization	103
5.3.3	Applications in graph clustering	108
5.3.4	Applications in graph Classification	110
5.3.5	Computational cost	112
5.4	Conclusions and perspectives	114
6	Schizophrenia prediction in the human connectome	117
6.1	Introduction	118
6.1.1	Machine learning and biomarker selection	119
6.2	Dataset and methods	120
6.2.1	Dataset	120

6.2.2	Biomarker evaluation protocol	121
6.2.3	Assessing the stability of feature selection	122
6.2.4	Measuring the classification performance	123
6.2.5	Embedded feature selection (RFE-SVM)	123
6.3	Results	125
6.3.1	Connectomes classification and features stability	125
6.3.2	Identification of brain regions in schizophrenia	126
6.3.3	Characterizing intergroup differences	128
6.4	Conclusions and perspectives	129
7	Conclusions and perspectives	131
	Appendices	135
A	Brain data preprocessing	137
A.1	Subjects	137
A.2	Brain network estimation	137
B	Statistical tests	141
B.1	Friedman-Nemenyi test	141
B.2	Mann-Whitney-Wilcoxon (MWW) test	142
B.3	Pearson correlation coefficient	143
C	Parameters of experiments	145
C.1	Multi-hop assortativities	145
C.2	Unsupervised Network Embeddings	145
C.2.1	Denoising autoencoder parameters	145
C.2.2	Graph2vec parameters	146
C.2.3	GAM parameters	146
C.2.4	Subgraph2vec parameters	146
C.2.5	Synthetic datasets	146
D	Datasets	149
D.1	Anomaly detection on attributed networks	149
D.2	Multi-hop assortativities in networks	149
D.3	Network embeddings with node identity awareness	149
	Bibliography	151
	Index	170

List of Figures

1.1	Outline of the thesis	21
2.1	The seven bridges of Königsberg	25
2.2	Markov stability example	35
2.3	The Iris dataset	37
2.4	The undercomplete autoencoder	44
3.1	Anomaly detection on attributed networks	47
3.2	A toy example of a work relation network	49
3.3	Anomaly detection on synthetic social network (income)	55
3.4	Performance of anomaly detection on synthetic networks (ROC-AUC)	60
3.5	Performance of anomaly detection on synthetic networks (PR-AUC)	60
3.6	Running time for computing the exponential of the Laplacian	63
3.7	Anomaly detection on the Disney network dataset	64
4.1	Assortativity on a US school friendship network	69
4.2	Multi-hop assortativity for graph classification	71
4.3	Autocovariance of node attributes (assortativity)	74
4.4	Eigenvectors as network features	77
4.5	Autocovariances as network features	77
4.6	Friedman-Nemenyi results on chemoinformatics	86
4.7	Friedman-Nemenyi results on social networks	86
5.1	Embedding a collection of networks	93
5.2	Unsupervised network embeddings overview	95
5.3	Denoising autoencoder	99
5.4	Visualization of Erdős-Rényi networks	104
5.5	Visualization of ER/BA networks and LFR networks	105

5.6	Visualization of LFR networks	106
5.7	Visualization of primary school temporal network	107
5.8	Dendrogram for the European airports multiplex network . . .	111
5.9	Running time for derivating distance matrix between networks	113
6.1	Structural connectome (brain network)	117
6.2	Overview of biomarker selection for schizophrenia predicition . .	121
6.3	Classification accuracy and stability for structural connectomes	124
6.4	Classification accuracy and stability for functional connectomes	124
6.5	Classification accuracy and stability for multimodal connectomes	125
6.6	Accuracy versus stability of features for all brain modalities . .	126
6.7	Degree of relevance for ROIs in the structural connectome . . .	127
6.8	Degree of relevance for ROIs in the functional connectome . . .	128
6.9	Brain surface maps for schizophrenia	128
6.10	Connectivity strength within and outside the a-core	129
A.1	Extraction of brain networks from empirical data	138

List of Tables

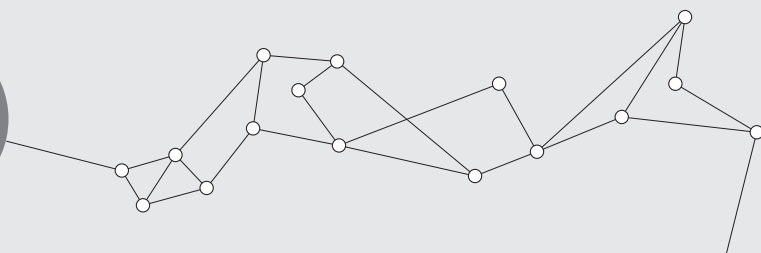
3.1	Real life benchmark for anomaly detection on networks	59
3.2	Anomaly detection results on real life datasets	62
4.1	Chemoinformatic and social network benchmarks for network classification	80
4.2	Selected features for graph classification	83
4.3	Classification results with graph-kernel methods on social networks	84
4.4	Classification results with neural nets and feature-based approaches on social networks	84
4.5	Classification results of graph-kernel methods on chemoinfor- matic networks	84
4.6	Classification results of RW kernel methods on chemoinformatic networks	85
4.7	Classification results of deep graph-kernel and neural nets ap- proaches on chemoinformatics	85
4.8	Running time for multi-hop assortativity computation	88
5.1	Synthetic dataset for network clustering and visualization	102
5.2	Clustering results for graph distances	109
5.3	Clustering results for graph embeddings	109
5.4	Classification results of brain networks 1 (gender)	112
5.5	Classification results of brain networks 2 (gender)	112

Notations and Acronyms

For convention, all matrices and vectors are in **bold**.

$\mathcal{D}$	Dataset
$\mathcal{G}$	A graph
$\mathcal{V}$	Set of vertices of a graph
$\mathcal{E}$	Set of edges of a graph
N	Number of nodes of a graph
m	Number of edges of a graph
$\mathbf{1}$	Vector of ones, i.e. all entries equal to 1
$\mathbf{d}$	Vector of the node degrees (d_u is the degree of the node u)
$\langle d \rangle$	Average degree
$\mathbf{D}$	Diagonal degree/strength matrix with $\mathbf{d}$ as diagonal
$\mathbf{A}$	Adjacency matrix of a graph
$\mathbf{W}$	Weighted adjacency matrix of a graph
$\mathcal{L}$	Combinatorial Laplacian matrix of a graph
$\mathbf{M}^t$	t -step transition matrix associated to a Markov process
Q	Modularity
r	Assortativity mixing coefficient
$\hat{f}$	Graph Fourier transform of a graph signal $\mathbf{f}$
$g(\mathcal{L})$	Graph filter or kernel applied on the spectrum of $\mathcal{L}$
δ_a	Kronecker delta centered in vertex a
$\mathbf{p}(t)$	Continuous time density vector ($p_i(t)$ is the probability that a random walker is in the node i at time t)
π	Stationary distribution of an associated Markov process
$\mathbf{\Pi}$	Diagonal matrix with π as diagonal
$\mathbf{H}$	Partition indicator matrix

$\mathbf{R}(t; \mathbf{H})$	Clustered Autocovariance matrix
$r(t; \mathbf{H})$	Markov stability
$VI(t)$	Variation of information between partitions at time t
$VI(t, t')$	Variation of information between partitions within the time interval t and t'
$c_a(t)$	Concentration of ‘energy’ on a node a at time t
$\mathbf{X}_t$	Random node indicator vector at time t
$\mathbf{p}_t$	Discrete time probability distribution of a random walker at time t
$\boldsymbol{\rho}(t)$	Autocovariance matrix of a random walker at time t
$r(t, \mathbf{v})$	t – hop assortativity for the node’s attribute vector $\mathbf{v}$
$r(t, \mathbf{H})$	t – hop assortativity for the categorical node attribute matrix $\mathbf{H}$
SVM	Support vector machines
RF	Random forest
RFE-SVM	Recursive feature elimination Support vector machines
$\mathbf{A}^r$	r –power of the adjacency matrix of a graph
d_H	Hamming distance between two matrices
d_J	Jaccard distance between two matrices
d_{DC}	DeltaCon similarity between two matrices
d_{Emb}	Network embedding distance between matrices
NMI	Normalized mutual information between two partitions
DTRW	Discrete time random walk
CTRW	Continuous time random walk
ML	Machine learning
ER	Erdős-Rényi networks
BA	Barabási-Albert networks
Mixed	ER and BA networks
LFR	Lancichinetti–Fortunato–Radicchi networks
DAE	Denosing autoencoder
MWW	Mann-Whitney-Wilcoxon hypothesis test
ROIs	Regions of interest (brain networks)
gFA	Generalized fractional anisotropy
iADC	Inverse apparent diffusion coefficient



Context and motivation

The advances in science and technology over the last decades have allowed generating, storing, and processing an unprecedented amount of data, offering new opportunities and challenges for studying complex systems. In this revolution, the fields of network science and machine learning have exploded, becoming increasingly important in the research community. Moreover, they have become a natural framework for applications in social networks [113, 149, 186], neuroscience [47, 153, 172], genomics [1, 115], finance [89, 116], medicine [41], and so on.

Network science has emerged during the first decade of the 21st century as a unified field [12], focused on the study of complex networks, e.g., group of agents interacting with each other. Networks arise in many real-life situations such as social networks, co-purchasing networks, communication networks, chemical compound networks, brain networks, and so on. In the areas of data mining, machine learning, information retrieval, and others, networks are modelled as graphs. A graph is a mathematical structure that consists of nodes also called vertices and links between nodes called edges. Networks can be defined as graphs where nodes and edges may have attributes (meta-data), e.g., gender or age in a social network. Networks from different areas can be represented with the same underlying graph. The mathematical theory studying graphs is called graph theory, and some of its generalities will be presented in Chapter 2.

One of the main tools in complex network analysis is the random walks (RWs) [8, 92]. RWs are at the core of several algorithms to uncover hidden structures on networks. For instance, centrality measures on networks such as PageRank [107], diffusion maps for dimensionality reduction of high-dimensional data [43], node-level similarity for collaborative recommendation [62], community detection based on diffusion flow trapped within certain regions of the

network [50, 106, 157] are prominent applications of RWs. Random walks are often used as a model of diffusion with different formulations and interpretations differing mainly in the nature of the time domain [126]. Understanding the interplay between dynamics and network structure, e.g., how the structure drives the dynamics and how the dynamics reveal hidden structure on networks is a crucial issue for the applications.

On the other hand, machine learning as a sub-field of artificial intelligence (AI) seeks to design systems with the ability to learn and improve from experience without being programmed explicitly. The experience is acquired from data through a learning process that allows the machines to solve non-trivial tasks involving perception, pattern recognition, and prediction. For instance, an important task in neuroscience is the diagnosis of neural disorders such as schizophrenia [75, 121] (see Chapter 6). After preparing a dataset of brain measurements i.e., fMRI scans, of both healthy and ill subjects, a machine learning algorithm will learn to discriminate between these two states. Thereby, it will be able to diagnose the neural state of a subject correctly. The fact of providing beforehand the current mental state of each subject to the learning algorithm is known as *supervised learning*.

Despite the large volume of available data, in many real situations, it is difficult or sometimes impossible to have prior knowledge about the expected output of the task, making it *unsupervised*. Consider the problem of community detection in real-life networks [68, 140], i.e., finding groups of densely connected nodes with few edges between groups. As the usual mechanism of communities generation is generally unknown in real systems, the ground-truth output is unknown [146]. Thus, unsupervised methods aim to uncover the intrinsic structure in the data relying on similarity relations between data entities.

With the emergence of suitable datasets and the need for more accurate results, many network related problems have recently raised the attention of the machine learning community and vice versa ¹. Nevertheless, this poses several challenges in the use of traditional machine learning methods on networks:

- *Inter-dependent nature of the data.* Networks reflect dependencies between data entities. Furthermore, data mining and machine learning algorithms on networks require to operate not only on individual entities but also consider their interactions.

¹The international conference in network science (Netsci) opened its first satellite in Machine Learning Applications in Network Science from the 2017 edition.

RWs have shown to be a helpful tool in network analysis for the discovery of recurrent patterns and relationships between data. A crucial question that arises in that direction is how diffusion processes on networks such as RWs, can reveal unusual structures (behaviour) on networks?

- *Representation of graph-based data.* Most machine learning algorithms have been developed to operate on vectorized data. Therefore, finding effective ways to encode graph structures in such a way that it can be used in traditional machine learning tasks, such as graph classification, clustering, visualization is one of the challenges we address in this thesis.
- *Feature extraction and selection on networks.* The choice of graph features is critical for machine learning applications, e.g., graph classification (see Chapter 4). The question of either learning features automatically, or hand-engineering domain-specific features for networks is an important concern we address in this work.

In this thesis, we investigate specific machine learning/mining applications in complex networks in the lines mentioned above. The focus of the thesis inquires into the developing of algorithms needed to perform supervised and unsupervised machine learning in graph-shaped data. We design the necessary tools for network analysis, performing rigorous experiments in synthetic and real-life data, interpreting the results with a critical eye, and drawing objective conclusions about the phenomenon under study.

Contributions and structure of the thesis

The primary purpose of the present thesis is to bring together two broad topics: machine learning and complex networks. We develop key applications employing RW dynamics on networks as a proxy for bridging the gap between these topics. Furthermore, feature engineering on graph-based data remains as the backbone of this work. Besides, we make a distinction in problems related to a single network, and others involving a collection of networks.

The structure and contributions of this thesis are summarized in the following:

We introduce in Chapter 2 some general concepts on graph theory and supervised/unsupervised machine learning that are necessary for the next chapters. Also, some supervised and unsupervised learning algorithms are described as used in further chapters. The first contribution of this thesis pertains to

mining a single network. As will be presented in Chapter 3, we propose an algorithm to perform anomaly detection in attributed networks. We characterize anomalous nodes as vertices having a highly concentrated ‘energy’ as a result of smoothing a unit impulse signal centred on each vertex with a low-pass filter. Indeed we will see that it can be interpreted in terms of diffusion of heat propagating across the network. Because anomalous nodes require a notion of context to evaluate the degree of disagreement concerning the surrounding nodes, they are intrinsically related to the graph clustering problem. This chapter makes the following contributions:

- A new multi-scale anomaly detection algorithm on attributed networks based on heat diffusion on networks and graph signal processing tools.
- An efficient way to compute the stability of a graph partition [50] linking diffusion and filtering of graph signals with Chebyshev polynomials.

In Chapter 4, we propose a generalization of assortativity mixing [137] in networks. We define multi-hop assortativities as covariances between nodes attributes at extremities of random paths of varying length. Assortativity based features offer a versatile family of fingerprints to characterize networks. Therefore, we propose assortativity based features from a collection of graphs and use them as descriptors to perform supervised graph classification. This chapter makes the following contributions:

- A generalization of assortativity mixing on networks incorporating categorical/scalar node attributes.
- A new way to build graph-based feature vectors based on multi-hop assortativity features to perform accurate graph classification.

In contrast, in Chapter 5, we provide an unsupervised approach to learn network embeddings from a family of graphs defined on the same set of nodes. We perform several numerical experiments on synthetic and real-life data with applications in graph visualization, clustering, and classification. This chapter makes the following contributions:

- A new methodological neural network approach to learning vector representations of graphs defined on the same set of nodes.

Finally, the last contribution relates to the identification of stable biomarkers for schizophrenia diagnosis in the human connectome (brain networks).

As will be presented in Chapter 6, we perform a systematic feature selection process on structural, functional, and multi-modal connectomes in order to identify a meaningful subset of markers for the diagnosis of schizophrenia. Besides, assessing the robustness of the biomarkers allowing us to identify the affected core in schizophrenia. This chapter makes the following contributions:

- A methodological approach to perform a stability analysis of feature selection algorithms applied on human brain connectivity networks.
- Identification of a set of biomarkers, i.e., brain regions, involved in schizophrenia diagnosis.

As the chapters explore a range of different topics, most of the chapters are self-contained. However, many of them have overlapping topics and applications. To facilitate the navigation between different chapters, a conceptual diagram is provided below.

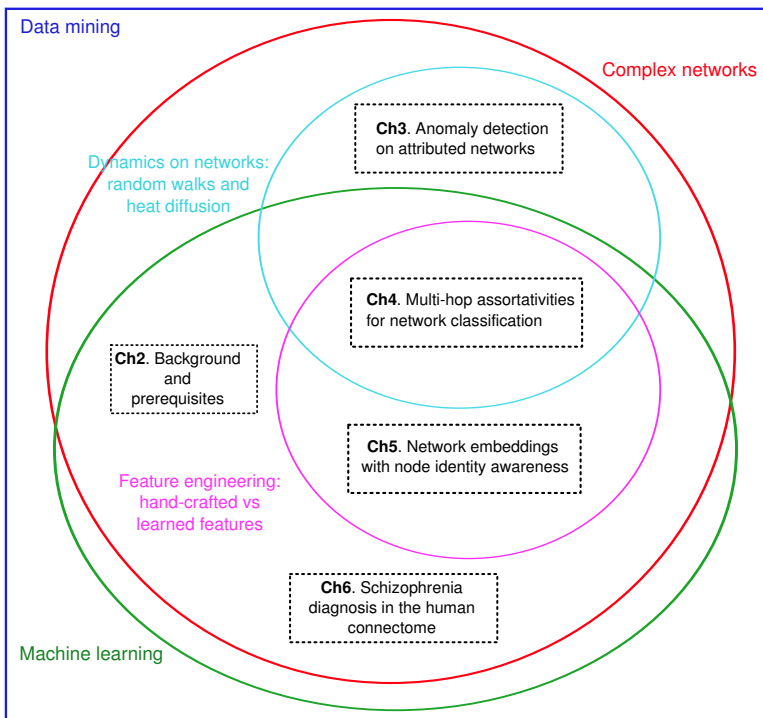


Figure 1.1: A brief overview of the main topics considered in the core chapters of the thesis. Colored text is the caption of colored shapes.

Publications

The production of this thesis has led some papers which either have been already published or submitted for publication. All papers were written in collaboration with Prof. Jean-Charles Delvenne. Some papers have been written with other collaborators who are mentioned in the following.

Peer reviewed international journals

Published

- Leonardo Gutiérrez-Gómez, Jean-Charles Delvenne
Multi-hop assortativities for network classification, Journal of Complex Networks, cny034, <https://doi.org/10.1093/comnet/cny034>
- Leonardo Gutiérrez-Gómez, Jean-Charles Delvenne
Unsupervised network embedding with node identity awareness, Applied Network Science Journal, special issue machine learning with graphs. Preprint arxiv:1903.05980

Under review

- Leonardo Gutiérrez-Gómez, Jakub Vohryzek, Benjamin Chiêm, Philipp S. Baumann, Philippe Conus, Kim Do Cuenod, Patric Hagmann and Jean-Charles Delvenne, *Stable Biomarker Identification For Predicting Schizophrenia in the Human Connectome*, NeuroImage: Clinical Journal. Preprint biorxiv doi: <https://doi.org/10.1101/711135>

Results of Chapter 6 are based in the above paper and were obtained in collaboration with:

- Jakub Vohryzek, Department of Psychiatry, University of Oxford, Oxford, United Kingdom.
- Benjamin Chiêm, ICTEAM, Université catholique de Louvain, Louvain-la-Neuve, Belgium.
- Philipp S. Baumann, Service of General Psychiatry and Center for Psychiatric Neuroscience, Department of Psychiatry, Lausanne University Hospital (CHUV), Lausanne, Switzerland
- Philippe Conus, Service of General Psychiatry and Center for Psychiatric Neuroscience, Department of Psychiatry, Lausanne University Hospital (CHUV), Lausanne, Switzerland

- Kim Do Cuenod, Service of General Psychiatry and Center for Psychiatric Neuroscience, Department of Psychiatry, Lausanne University Hospital (CHUV), Lausanne, Switzerland
- Patric Hagmann, Department of Radiology, University Hospital Centre and University of Lausanne, Lausanne, Switzerland
- Jean-Charles Delvenne, ICTEAM, CORE, Université catholique de Louvain, Louvain-la-Neuve, Belgium.

Peer reviewed international conferences

Under review

- Leonardo Gutiérrez-Gómez, Alexandre Bovet, Jean-Charles Delvenne, *Multi-scale anomaly detection on attributed networks*, AAAI-20 Conference on Artificial Intelligence.

Results of Chapter 3 are based in this paper, and were obtained in collaboration with Alexandre Bovet from ICTEAM, Université catholique de Louvain, Louvain-la-Neuve, Belgium.

Other international conferences

Published

- Leonardo Gutiérrez-Gómez, Jean-Charles Delvenne *Network Embeddings for Graph Classification and Visualization*, The 7th International Conference in Complex Networks and their Applications, 2019. Book of abstracts.

2

Background and prerequisites

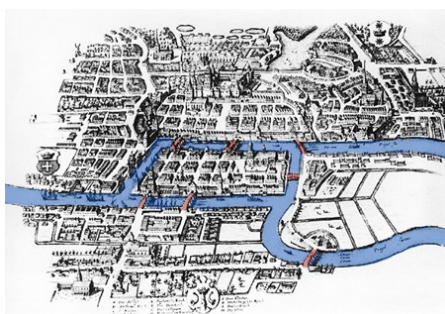


Figure 2.1: The seven bridges of Königsberg.

In this chapter, we give a short introduction of some background material that will be the prerequisites we build upon later. A general introduction to complex networks is given in terms of spectral graph theory and assortativity mixing on networks. Subsequently, we review community detection, random walks and the Markov stability framework. We described supervised and unsupervised machine learning, feature selection and some representative algorithms we use later on. We recommend the expert reader to skip this chapter and refers to it only if needed.

Introduction to complex networks

Graph theory

The origin of graph theory has been attributed historically to the solution of the Königsberg bridge problem. Given the seven bridges especially arranged as in Figure 2.1, is there any way to walk across all of them without cross the same twice and come back to the initial point?. The problem remained without a rigorous mathematical proof until 1735, when Leonhard Euler came

up with a solution proving that such a walk does not exist [57]. He turned the problem in a graph representing each of the four land areas from the figure as a node and bridges connecting lands as edges; then he proved that such a path could not exist on a graph with nodes having an odd number of links [57]. The take-home message from this problem is simple: the problem became simpler when it is modelled as a graph, and the fact that such a walk does not exist is an intrinsic property of that graph.

In mathematics and computer science, graph theory seeks the study of graphs, where a graph is a collection of nodes and edges connecting pairs of nodes. Although in this chapter we present the basis of graph theory and some tools for analyze them, we refer the reader to the textbooks [12, 15, 74, 136] for a broader description of the topic.

More formally, a graph is an ordered pair $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ comprised by a set of nodes (or vertices) $\mathcal{V}$, and a set of links (or edges) $\mathcal{E}$ connecting pairs of nodes. When edges do not have an explicit direction, the graph is *undirected*; otherwise, it is known as *directed*. The size of $\mathcal{V}$ and $\mathcal{E}$ can be infinite, but as we will develop some applications, we will consider the finite case, with the number of vertices $|\mathcal{V}| = N$ and edges $|\mathcal{E}| = m$.

A graph can be represented by its *adjacency matrix*, a square $N \times N$ matrix $\mathbf{A}$ encoding the connectivity between nodes, where the $A_{i,j}$ entry equal to 1 if nodes i and j are connected by an edge and 0 otherwise.

In some applications, it is necessary to assign weights on the edges of a graph. For instance, in a mobile call network [17], weights might represent the number of calls between users, or the time spent during a call. In this case, a graph can be represented by its *weighted adjacency matrix* $\mathbf{W}$, an $N \times N$ matrix with the entry $W_{i,j} \in \mathbb{R}$ representing the weight associated to the edge (i, j) .

The number of out-going edges from a vertex $v \in \mathcal{V}$ is known as the *degree* of the node and is denoted by d_v . Sometimes is convenient to represent the *degree matrix* of a network as the diagonal matrix $\mathbf{D} = \text{diag}(\mathbf{A}\mathbf{1})$, where $\mathbf{1}$ is an $N \times 1$ vector of ones. In the case of directed graphs, the number of out-going edges from $v \in \mathcal{V}$ is known as the *out-degree* of v and the number of edges arriving to v is known as the *in-degree* of v . When the graph is undirected, its adjacency matrix is symmetric having a maximum number of edges $m = \frac{N(N-1)}{2}$, whereas in the directed case $m = \sum_{i,j} A_{i,j}$.

The number of edges is very important for practical reasons. For instance, it is well known that social networks, e.g., a graph with nodes representing people

and edges social ties, are sparse as few entries of its adjacency matrix are equal to one. As a consequence, storing large networks or operating on them is more efficient in time and memory. We will show an example in Chapter 3 exploiting the sparsity of the network to improve running time.

The graph Laplacian

We previously introduced the adjacency matrix capturing the entire structure of a graph. However, there are several alternative ways to represent a network. For instance, the modularity matrix, random walk matrix or the graph Laplacian, etc. Among them, the graph Laplacian is the most widely used.

The *graph Laplacian* [39, 40] of an undirected, unweighed network with N nodes has elements:

$$\mathcal{L}_{i,j} = \begin{cases} d_i, & i = j \\ -1, & i \neq j, (i,j) \in \mathcal{E} \\ 0, & \text{otherwise} \end{cases} \quad (2.1)$$

where d_i is the degree of node i . It can be written in matrix form as:

$$\mathcal{L} = \mathbf{D} - \mathbf{A} \quad (2.2)$$

One can also write the Laplacian for a weighted matrix, replacing above the degree matrix by $\mathbf{D} = \text{diag}(\mathbf{W}\mathbf{1})$ and the adjacency matrix for $\mathbf{W}$.

The eigenvalues of the Laplacian are of particular interest for understanding the spectral properties of a graph. Since the Laplacian is real symmetric matrix, it has non-negative eigenvalues. Indeed, let λ be any eigenvalue of $\mathcal{L}$ and $\mathbf{v}$ its corresponding eigenvector unit normalized so that $\mathbf{v}^T \mathbf{v} = 1$. It turns out that

$$\mathbf{v}^T \mathcal{L} \mathbf{v} = \lambda \quad (2.3)$$

and on the other hand,

$$\mathbf{v}^T \mathcal{L} \mathbf{v} = \sum_{(i,j) \in \mathcal{E}} A_{i,j} (v_i - v_j)^2 \geq 0 \quad (2.4)$$

which actually proves that the graph Laplacian is a *positive semi-definite matrix*.

As the rows (and columns) of $\mathcal{L}$ sum to zero, $\mathcal{L}\mathbf{1} = 0$ then the vector of ones is always an eigenvector of $\mathcal{L}$ with eigenvalue zero. It can be proved that the

number of zero eigenvalues of the Laplacian are associated with the number of disconnected components of the graph [40].

We can then sort the eigenvalues of $\mathcal{L}$ as $0 = \lambda_0 < \lambda_1 \leq \dots \leq \lambda_{N-1}$, and represent its orthogonal eigenvectors in a matrix form $\mathbf{U} = [\mathbf{u}_0, \dots, \mathbf{u}_{N-1}]$. Hence, the graph Laplacian can be written as:

$$\mathcal{L} = \mathbf{U} \mathbf{\Lambda} \mathbf{U}^T \quad (2.5)$$

where $\mathbf{\Lambda}$ is a diagonal matrix with the eigenvalues of $\mathcal{L}$ in its diagonal.

Assortativity mixing

Real systems are characterized by the heterogeneity in the way their nodes interact, forming structural patterns. One way to characterize the network structure is by looking at the correlations between node attributes (or meta-data). This property known as *assortativity mixing* or simply *assortativity* [137] allows us to make judgements concerning the presence of edges between nodes with similar characteristics (assortativity), or between those with different ones (disassortative). For instance, assortativity has been widely studied in social networks (known as homophily) observing attributes such as degree, race or gender [97], [110], etc.

Assortativity mixing is often defined for numerical characteristics such as age or weight in social networks, item price or the number of reviews in a co-purchasing network. It quantifies the tendency of nodes with similar values of a scalar characteristic to be linked, i.e., people with similar age tend to create friendship ties. The assortativity between scalar node attributes is quantified as a covariance. Consider the process of selecting uniformly at random (or proportionally to the weight if any) edges on a network. The values at the extremities of a selected edge, i.e., node attributes, are random variables. Thus, the covariance between scalar node attributes x_i and x_j of nodes i and j reads [137]:

$$\text{cov}(x_i, x_j) = \frac{1}{2m} \sum_{i,j}^N \left(A_{i,j} - \frac{d_i d_j}{2m} \right) x_i x_j \quad (2.6)$$

It is sometimes convenient to normalize the covariance so that it takes the value of 1 for perfectly assortative networks, and -1 for disassortative ones. The value of 0 indicates that the values at the extremities of the selected edge

are uncorrelated. This normalized measure is called the *assortativity coefficient* and can be interpreted as a Pearson correlation (see appendix B.3) in the form:

$$r = \frac{\text{cov}(x_i, x_j)}{\sigma_i \sigma_j} = \frac{\sum_{ij} A_{ij} (x_i - \bar{x})(x_j - \bar{x})}{\sum_i d_i (x_i - \bar{x})^2} \quad (2.7)$$

where, $\bar{x} = \frac{1}{2m} \sum_i d_i x_i$ is the mean value over all node attributes weighed by the node degree, and σ_i is the standard deviation of the attribute values. In the context of spatial statistics, this expression is known as the Moran and Geary indexes [38, 86] commonly used to quantify autocorrelation of a single node feature with the graph structure.

Assortativity mixing can also be defined for categorical attributes such as race or gender in social networks. In mathematical terms, let g_i be the group or type of node i , which is encoded as an integer $g_i \in \{1, \dots, K\}$ with K being the number total of groups, i.e. if the node type is gender $g_i \in \{\text{male}, \text{female}, \text{other}\}$. The Kronecker delta δ_{g_i, g_j} is equal to 1 if the type (or group) of the nodes i and j is the same, and zero otherwise. The quantity:

$$Q = \frac{1}{2m} \sum_{i,j}^N \left(A_{i,j} - \frac{d_i d_j}{2m} \right) \delta_{g_i, g_j} \quad (2.8)$$

is known as *modularity* [137, 140] when the node attribute encodes node group membership in community detection [140]. In the context of categorical node attributes, Q informs about the *assortativity mixing* of a network. This quantity measures the fraction of edges between nodes of the same type (or group) minus the expected number of edges between all pairs of nodes of the same type we would expect by chance. Modularity has a maximum value of 1 if there are more edges between nodes of the same type than we would expect by chance. It can also be negative when there are fewer such edges than we would expect by chance. For example, heterosexual dating networks (gender), ecological food webs (metabolic category) are examples of networks with negative modularity.

In Chapter 4, we develop a generalization of assortativity mixing as covariances between scalar or categorical node attributes in different path lengths (t -hops). In this way, the aforementioned definition turns into 1-hop assortativity under our framework.

Community detection on networks

The ultimate goal of studying complex networks is to understand better the underlying system they represent. Related with the computer science strategy,

divide and conquer, community detection refers to divide the whole network into interrelated components in order to gain insights about the complete system. Community detection is defined often as the task of finding a partition of the network into groups of nodes such that there are many edges within each group and few edges between them [140]. However, this definition is vague and susceptible to different interpretations allowing the emergence of many community detection methods. For instance, *spectral methods* [59] are based on the spectral properties of the graph Laplacian. *Modularity based methods* [7, 100, 137, 140] stated the community detection problem as an optimization problem in terms of the modularity measure, Eq. 2.8. Methods using *information theory* [178] stated the problem as a data compression problem. *Dynamical based methods* [51, 56, 106] aim to uncover the modular structure of the network employing random walks and other diffusion processes.

Although in this thesis we do not tackle the problem of community detection directly, we borrow some essential elements from the dynamical-based paradigm to build some of our contributions.

Random walks in networks

One of the fundamental differences between graphs and networks is that networks involve not only the study of static structures but also the dynamical processes taking place on the underlying graph. For this reason, we review a fundamental dynamical process on networks that is the basis of several of our contributions: *random walks*.

Discrete time random walks (DTRW)

The DTRW [28, 99] considers a walker localized on one node of a graph moving at each time step to one of its direct neighbors at random. When the walker is at a node v_i it moves to one of its neighbors v_j with a probability proportional to the edge (i, j) . These probabilities are encoded in an $N \times N$ transition matrix $\mathbf{M}$, where the entry $M_{i,j}$ gives the probability that a walker moves from the node v_i to v_j , and is computed as:

$$M_{i,j} = \frac{A_{i,j}}{d_i} \quad (2.9)$$

with d_i as the degree of the node v_i . It can be written in a matrix form as $\mathbf{M} = \mathbf{D}^{-1}\mathbf{A}$. Although there are multiple ways to define $\mathbf{M}$ (see [28]), here we consider the simple random walk with uniform transition probability. Thus,

the transition probability distribution evolves as:

$$\mathbf{p}_{n+1} = \mathbf{p}_n \mathbf{M}, \quad \text{and} \quad \sum_{i=1}^N p_n(i) = 1 \quad (2.10)$$

with $p_n(i)$ as the probability of a walker to be at node v_i at discrete time n . Hence, the probability vector at time n can be written as $\mathbf{p}_n = \mathbf{p}_0 \mathbf{M}^n$.

Considering an undirected, connected and non bipartite graph, the dynamical process converges [28] to the *stationary density* vector $\boldsymbol{\pi} = (\pi_1, \dots, \pi_N)$, where $\pi_i = \lim_{n \rightarrow \infty} p_n(i)$. Thus, the stationary distribution satisfies:

$$\boldsymbol{\pi} = \boldsymbol{\pi} \mathbf{M}, \quad \text{with} \quad \sum_{i=1}^N \pi_i = 1 \quad (2.11)$$

It can be seen that the stationary density can be obtained by computing the left eigenvector of $\mathbf{M}$ associated to the eigenvalue 1. It turns out that, under the above mentioned hypothesis, the stationary density of the random walk is proportional to the degree of the nodes [154]. Indeed, starting from $\mathbf{M} = \mathbf{D}^{-1} \mathbf{A}$, the right-hand side of Eq. 2.11 becomes $\boldsymbol{\pi} \mathbf{M} = \boldsymbol{\pi} \mathbf{D}^{-1} \mathbf{A}$ and setting $\boldsymbol{\pi} = \mathbf{d}^T$,

$$\mathbf{d}^T \mathbf{M} = \mathbf{d}^T \mathbf{D}^{-1} \mathbf{A} = \mathbf{1}^T \mathbf{A} = \mathbf{d}^T \quad (2.12)$$

Thus, the node degree vector $\mathbf{d}^T$ is an unnormalized left-eigenvector of $\mathbf{M}$ associated with the eigenvalue 1. Finally, normalizing the probability density vector:

$$\pi_i = \frac{d_i}{2m}, \quad \text{for all } 1 \leq i \leq N. \quad (2.13)$$

For a broader presentation of random walks, Markov chains and more general stochastic processes, we refer the reader to the references [28, 99, 143].

Continuous time random walk (CTRW)

The CTRWs [143] are characterized by having two main components: information of a walker's trajectory in terms of the number of steps, and statistics of the times at which events take place. Combining these two components, one can specify the probability that a random walker will visit a given node in a given time. In CTRWs we can distinguish two instances [126], depending on whether the dynamics take place at a node or edge levels. In this thesis, we consider the edge-centric continuous-time random walks (denoted as CTRW for simplicity), as a more appropriated dynamics for our applications. Consider a connected, undirected and non-bipartite graph. The CTRW assumes that each

edge is activated independently according to a renewal process such that when the edge is activated then the walker can move to the adjacent node. Consider a Poisson process defined independently to each edge with a rate proportional to its weight. Using an inhomogeneous rescaling of time as [106], this leads to the master equation:

$$\frac{d\mathbf{p}(t)}{dt}\langle\mathbf{d}\rangle = \mathbf{p}(t)(-\mathbf{D} + \mathbf{A}) = -\frac{\mathbf{p}(t)}{\langle\mathbf{d}\rangle}\mathcal{L} \quad (2.14)$$

where $\mathbf{A}$ is the adjacency matrix of the graph, $\mathbf{D}$ the diagonal matrix of node degrees, i.e., $\mathbf{D} = \text{diag}(\mathbf{d})$, and $\langle\mathbf{d}\rangle$ is the nodes average degree. The dynamics is driven by the $N \times N$ combinatorial Laplacian matrix $\mathcal{L}$. This equation implies that the waiting time for a walker at each node is inversely proportional to its degree or strength, i.e., a walker will leave faster from a node with a high degree or strength. The Eq. 2.14 is then solved by:

$$\mathbf{p}(t) = \mathbf{p}(0)e^{-\frac{t}{\langle\mathbf{d}\rangle}\mathcal{L}} \quad (2.15)$$

Furthermore, in the limit of large times the dynamics converges [28] to the stationary density vector $\boldsymbol{\pi}$ satisfying:

$$\boldsymbol{\pi}\mathcal{L} = \mathbf{0}. \quad (2.16)$$

That is, $\boldsymbol{\pi}$ is the left eigenvector of $\mathcal{L}$ with eigenvalue 0. Note that as the sum of the rows/columns of $\mathcal{L}$ are equal to zero, then $\boldsymbol{\pi} = \mathbf{1}^T$ satisfies the previous equation. Finally, normalizing the density vector:

$$\boldsymbol{\pi} = \frac{1}{N}(1, \dots, 1)^T. \quad (2.17)$$

In the next section we present the Markov stability framework for community detection, formulated in terms of discrete and continuous time random walks.

Introduction to the Markov stability framework

Real-world networks representing complex social, economic, biologic systems are characterized by modular, multi-scalar, and often hierarchical structures [19]. An important challenge in the analysis of such systems is the identification of meaningful components or communities, providing insights about the function of the global system. In that direction, the Markov stability framework [50, 106] provides a dynamical setting to unveil the mesoscopic structure of networks in a principled manner. Consider the diffusion of information in a

friendship social network. Intuitively, the information will be transferred from one person to its direct friends, and from them to their friends and so on, accordingly with the underlying friendship graph. In regions of the network with a high density of friendship ties, the information flow will remain trapped for longer times, before it goes out to another group until it has eventually passed throughout the whole network. By keeping track of the time dynamics of the diffusive process, the Markov stability framework allows uncovering the multi-scalar structure of networks.

Autocovariance of a graph partition

One of the main advantages of this framework is that it allows the incorporation of different dynamics targeted to reveal diverse aspects of the network structure. In this thesis, we consider the generic random walk defined on Eq. 2.10, and the continuous case presented in Eq. 2.14 as dynamical processes taking place on the network.

As was presented in the previous section, under the assumptions of an undirected, connected and non-bipartite graph, both dynamics converge to an unique $1 \times N$ stationary vector $\boldsymbol{\pi}$ accounting for the long term behaviour of the system. Consider the hard partition of the graph in K groups or clusters. The partition is encoded into a $N \times K$ binary matrix $\mathbf{H}$, with the entry $h_{i,j} = 1$ when the node i belongs to the cluster j . The *clustered autocovariance* matrix [50, 51, 106] of the diffusion process at time t is defined as:

$$R(t; \mathbf{H}) = \mathbf{H}^T [\boldsymbol{\Pi} \mathbf{P}(t) - \boldsymbol{\pi}^T \boldsymbol{\pi}] \mathbf{H} \quad (2.18)$$

where $\mathbf{P}(t)$ is either the t -step transition matrix of the discrete-time process $\mathbf{P}(t) = \mathbf{M}^t$, or the continuous-time dynamics $\mathbf{P}(t) = e^{-\frac{t}{\langle d \rangle} \mathcal{L}}$, and $\boldsymbol{\Pi} = \text{diag}(\boldsymbol{\pi})$, with $\boldsymbol{\pi}$ as the respective stationary density distribution, Eq. 2.13 and Eq. 2.17.

The autocovariance matrix is written as a difference between two terms. Indeed, each entry $[R(t; \mathbf{H})]_{i,j}$ can be interpreted in terms of a random walk perspective. This is nothing but the probability of a random walker to start in the community i at stationarity, and ending up at community j after time t , minus the probability of two independent walkers to be in communities i and j at stationarity. In this way, communities correspond to group of nodes where the probability distribution is more concentrated after t -steps than expected

in stationary. In other words, the term of interest is:

$$r(t; \mathbf{H}) = \text{trace}[R(t; \mathbf{H})]. \quad (2.19)$$

This term is known as the *stability of the partition* $\mathbf{H}$ [51] and is a useful tool to assess the quality of different clusterings over time. Thus, the best partition at a given time scale t is obtained by maximizing the stability:

$$H^* = \arg \max_{\mathbf{H}} r(t, \mathbf{H}) \quad (2.20)$$

where a high stability value indicates a better partition of the graph into communities at a given time t according with the desired dynamics. As the time increases, the random walker explores larger regions of the network, yielding with coarser partitions for large times. The optimization of the Markov stability results in a sequence of partitions optimal over different times. Although this optimization is known to be NP-hard [24], we will discuss appropriated heuristic methods to address this problem later.

Stability and modularity

The Markov stability for undirected graphs represents a general framework consolidating many well-known measures on networks. For an extensive presentation of them, we refer the reader to the references [50, 51]. The stability in Eq. 2.19 can be written as as:

$$r(t; \mathbf{H}) = \sum_{i,j=1}^N [\pi_i(\mathbf{P}(t))_{i,j} - \pi_i \pi_j] \delta_{c_i, c_j} \quad (2.21)$$

where $\delta_{c_i, c_j} = 1$ when nodes i and j are in the same cluster, and zero otherwise.

When considering the *discrete time* random walk dynamics (Eq. 2.10), the stability at time $t = 1$ yields the usual modularity with the configuration model as a null model [140]:

$$r(1; \mathbf{H}) = \sum_{i,j=1}^N \left(\frac{d_i}{2m} \frac{A_{i,j}}{d_i} - \frac{d_i}{2m} \frac{d_j}{2m} \right) \delta_{c_i, c_j} = \frac{1}{2m} \sum_{i,j=1}^N \left(A_{i,j} - \frac{d_i d_j}{2m} \right) \delta_{c_i, c_j} \quad (2.22)$$

Similarly, the modularity associated with the *continuous time* combinatorial Laplacian random walk (Eq. 2.14), is based on the Erdős-Rényi null model and defined for small times as [106]:

$$r(1, \mathbf{H}) = \text{trace} \left[\mathbf{H}^T \left(\frac{\mathbf{A}}{2m} - \frac{\mathbf{1}\mathbf{1}^T}{N^2} \right) \mathbf{H} \right] \quad (2.23)$$

It turns out that in general it is always possible to rewrite the Markov stability for any choice of random walk as a modularity of another symmetric graph [106]. This observations is useful in practice in the maximization of the stability because any heuristic developed to maximize a modularity function can be used, e.g., the Louvain algorithm [18].

Finding relevant scales

Once the sequence of optimal partitions is obtained from Eq. 2.20, we need to identify the relevant time scales, e.g., relevant partition for the network. As proposed in [106], the significance of a given partition is characterized by its robustness in two different ways.

First, the robustness of the partition with respect to the Louvain algorithm and random initializations, and second, the persistence of the partition across time. In any case, an optimal partition will be robust if after any perturbation its outcome change a little and is close to the unperturbed one. A popular way to compare two partitions P_1 and P_2 is with the normalized variation of

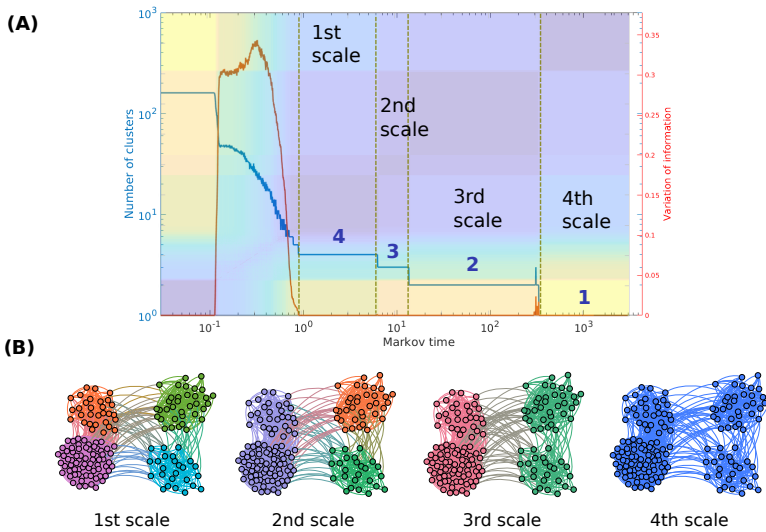


Figure 2.2: Consider a synthetic network with 160 nodes and a planted partition of four communities. (A) For a given Markov time t , we find the number of communities (blue curve) and compute the average variation of information of the retrieved partitions, after repeating the Louvain algorithm 100 times (red curve). The background represents the normalized variation of information between optimized partitions across times, i.e., $VI(P(t), P(t'))$. As can be seen, relevant scales emerge as block diagonal regions with low VI values (background) and continuous plateaux in the number of communities with low values in the $VI(t)$ (red curve). (B) A visualization of the optimal partitions retrieved at each relevant scale.

information (VI) [127]:

$$VI(P_1, P_2) = \frac{H(P_1|P_2) + H(P_2|P_1)}{\log(N)} \quad (2.24)$$

where $H(P_1|P_2)$ is the conditional entropy of P_1 given P_2 , i.e., the additional information needed to describe P_1 when P_2 is known assuming a uniform probability on the nodes, and is written as:

$$H(P_1|P_2) = - \sum_{x \in P_1, y \in P_2} p(x, y) \log \left(\frac{p(x, y)}{p(x)} \right) \quad (2.25)$$

where, $p(x)$ is the fraction of nodes in the cluster x and $p(x, y)$ the fraction of nodes in both clusters x and y . It can be proven that VI is a true metric and vanishes only when the two partitions are identical [127].

Within the Markov stability framework, this metric allows to identify persistent partitions across time. By looking at block diagonal region with low $VI(P(t), P(t'))$ in different times, as well as plateaux in the number of retrieved communities as a function of time, robust partitions can be detected. In addition, by repeating the Louvain algorithm many times for different initial values, a sequence of optimal partitions is obtained for any fixed time t . The robustness of the partition is then assessed by computing the average $VI(t)$ over the ensemble of solutions. Thus, a relevant partition will have a low value (or a dip) in the $VI(t)$ of the ensemble of partitions. We illustrate this in a toy example in Figure 2.2.

Introduction to machine learning

Machine learning (ML) is a subfield of artificial intelligence (AI) [69], where machines acquire their knowledge from data. This allows the machine to solve real problems involving some perceptual knowledge such that recognizing objects or concepts on images or videos, classifying songs, recommending whether a person needs surgical intervention, and so on.

In order to illustrate some relevant concepts, we will introduce a real-life workhorse dataset. The Iris dataset ¹ is one of the most used datasets studied in the machine learning community. It is composed of a collection of 150 examples of the iris plants. For each plant, four measurements were taken: the sepal length, sepal width, petal length, and petal width. Also, the species of

¹<https://archive.ics.uci.edu/ml/datasets/iris>

each plant was registered, e.g., Setosa, Vesicolor, and Virginica. See Figure 2.3 for an overview of the dataset.

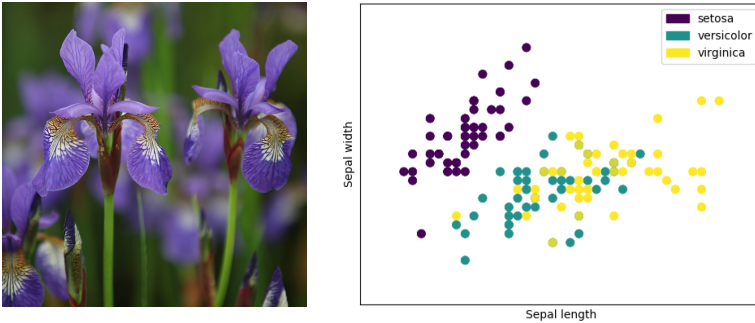


Figure 2.3: (Left) An Iris plant from the Setosa specie. (Right) Example of Iris dataset for two features. Colour of points indicate the plant specie.

ML addresses the problem of building automated systems capable of predicting the species of an iris plant, based on the previous measurements. These values are called features and describe individual data examples as relevant descriptors of the raw data object, see Figure 2.3. Features are determined through a feature selection process detailed in the following.

Feature selection and extraction

Machine learning often requires a pre-processing step for extracting or selecting features in order to perform classification or clustering tasks accurately. Features are individual measurements or characteristics describing the data. For instance, in Figure 2.3, the Sepal width and Sepal length are physical features extracted from the left-hand side plant allowing to discriminate fairly the three species. The reason for this is that raw data is complex, often noisy, and difficult to process without extracting the useful features beforehand.

Feature selection is the process of performing feature reduction, i.e., eliminating noise features, keeping those that best fulfil a selection criterion [147]. Existing feature selection algorithms can be organized into three categories [81]. The *filter* methods look at intrinsic properties of the data independently of the classifier design. On the contrary, *wrapper* methods assess the goodness of features using the performance of a predefined classifier. *Embedded* methods make use of the internal parameters of a learning algorithm to perform feature selection. For more details about feature selection, we refer the reader to Refs. [81, 82].

On the other hand, *feature extraction* is a form of dimensionality reduction where the raw data is transformed into derived values or features suitable for modelling. There exist globally two paradigms to perform feature extraction [29, 132]. The first one, known as *hand-crafted* feature engineering uses the user domain expertise of the data to create features to make ML algorithms work. This process is generally difficult and time-consuming. However, if well executed can yield incredibly powerful features optimized for particular ML problems. The second approach is known as *feature learning* or *representation learning*. The idea is to let the machine discover automatically the right features needed to perform any ML task. In ML tasks such as image recognition or classification, automatic feature learning with convolutional neural networks [104], turns out to be the best way to learn powerful features at the expense of losing in interpretation [132].

Supervised learning

Let us denote by $\mathbf{x}_i$ an input data point (an iris plant in our example), and y_i the output or label we want to predict for each example (the species of the plant). The pair $(\mathbf{x}_i, y_i)$ is known as a *training example* and the set of m examples $\{(\mathbf{x}_i, y_i), 1, \dots, m\}$ is called the *training set*. We denote as $\mathcal{X}$ the space of input examples and $\mathcal{Y}$ as the space of target labels. In *supervised learning* the goal is to learn a function $h : \mathcal{X} \rightarrow \mathcal{Y}$, such that for any $\mathbf{x}_i \in \mathcal{X}$, $h(\mathbf{x}_i)$ is a good predictor of $y_i \in \mathcal{Y}$. When y_i has discrete values, e.g., $y_i \in \{\textit{setosa}, \textit{versicolor}, \textit{virginica}\}$, it is known as a *classification problem*.

In the following, we introduce two supervised learning algorithms used in this thesis.

Support Vector Machine

Support Vector Machine (SVM) [23] is a powerful algorithm to perform classification or regression in supervised binary or multi-class prediction tasks. For simplicity, we will introduce the binary case. Let $\{(\mathbf{x}_i, y_i), 1, \dots, m\}$ be a training set with $\mathbf{x}_i \in \mathbb{R}^p$ and $y_i \in \{-1, 1\}$. The goal is to learn a mapping $f : \mathbb{R}^p \rightarrow \{-1, 1\}$ such that it can predict the output label accurately for any unseen data point.

In the case where the data is *linearly separable*, the decision about the right class of data points can be assessed with a hyperplane also known as the decision boundary, described by $\langle \mathbf{x}, \mathbf{w} \rangle + b = 0$, with $\mathbf{w} \in \mathbb{R}^p$ as a normal vector

and $b \in \mathbb{R}$. Hence, for a data example $\mathbf{x}_i$, the decision about its predicted label is done with:

$$f(\mathbf{x}_i) = \text{sign}(\langle \mathbf{x}_i, \mathbf{w} \rangle + b) \quad (2.26)$$

where $f(\mathbf{x}_i) \in \{1, -1\}$. The width of a band around the decision boundary without any training sample is known as the margin. Among all possible hyperplanes separating the data points, the optimal one is defined as the one which maximizes that margin.

This can be written as an optimization problem as:

$$\begin{aligned} & \underset{\mathbf{w}, b}{\text{minimize}} && \frac{1}{2} \|\mathbf{w}\|^2 \\ & \text{subject to} && y_i(\langle \mathbf{x}_i, \mathbf{w} \rangle + b) \geq 1, \quad \text{for all } i = 1, \dots, m. \end{aligned} \quad (2.27)$$

By the above equation, the margin is maximized and the constraints $y_i(\langle \mathbf{x}_i, \mathbf{w} \rangle + b) \geq 1$ indicate that points will lie outside that margin, minimizing the risk of misclassifying data.

Writing down the dual of Eq. 2.27, it turns into:

$$\begin{aligned} & \underset{\alpha}{\text{maximize}} && \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \\ & \text{subject to} && \sum_{i=1}^m \alpha_i y_i = 0, \quad \text{and } \alpha_i \geq 0 \quad \text{for all } i = 1, \dots, m. \end{aligned} \quad (2.28)$$

SVM is also extendible for cases where the data is *not fully linearly separable*. The so-called *soft-margin* SVM allows missing classification of data points in order to improve the generalization performance of the model. To do so, Cortez and Vapnik [44] introduced slack variables ξ and a regularization term penalizing high slack variables with a factor C . This formulation is known as the C -SVM. In its dual form, slack variables disappear, and the penalization factor becomes part of the constrain as follows:

$$\begin{aligned} & \underset{\alpha}{\text{maximize}} && \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \\ & \text{subject to} && \sum_{i=1}^m \alpha_i y_i = 0, \quad \text{and } 0 \leq \alpha_i \leq C \quad \text{for all } i = 1, \dots, m. \end{aligned} \quad (2.29)$$

yielding also:

$$\mathbf{w} = \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i, \quad \text{for all } i = 1, \dots, m. \quad (2.30)$$

This formulation has many advantages:

- Data points appear as dot products
- As well as the formulation 2.28, the objective is convex with linear constraints, which can be easily solved with any QP solver
- The coefficient vector $\mathbf{w}$ is written in terms of few data points, e.g., the vector $\boldsymbol{\alpha}$ is sparse, and provides a notion of features strength for the classification problem [81,83]. This property is exploited on Chapter 6 in the context of feature selection for schizophrenia prediction.

Support vector machine also has a *non-linear* formulation to deal with complex non-linear boundaries, i.e. quadratic, exponential, etc.

The idea here is to map the data points into a feature space where the problem becomes linearly separable. The challenge is to find that feature space and exploiting the fact that, as the Eq. 2.29, the data points are expressed as dot products. By mapping those points into an often, higher dimensional feature space, the product becomes:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle \quad (2.31)$$

where $\phi(\mathbf{x}_i)$ is the feature mapping of $\mathbf{x}_i$ in the high dimensional feature space and k is known as a *kernel mapping* [161]. Hence, during the *training phase*, the optimization becomes:

$$\begin{aligned} & \underset{\boldsymbol{\alpha}}{\text{maximize}} && \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) \\ & \text{subject to} && \sum_{i=1}^m \alpha_i y_i = 0, \quad \text{and } 0 \leq \alpha_i \leq C \quad \text{for all } i = 1, \dots, m. \end{aligned} \quad (2.32)$$

where C controls the trade-off between misclassification and the width of the margin.

Furthermore, during the *testing phase*, for a given unknown data point $\mathbf{x}_{test}$, its predicted label $\hat{y}_{test}$ reads:

$$\hat{y}_{test} = \text{sign} \left(\sum_{i=1}^m (\alpha_i y_i k(\mathbf{x}_i, \mathbf{x}_{test})) + b \right) \quad (2.33)$$

It can be noticed that during training and testing phases, all data points evaluation occurs as dot products between data examples in high dimensions. However, it is not required to know explicitly the mapping ϕ to evaluate the kernel. This is known as the *kernel trick*. Indeed, if we find a function k which is symmetric and positive semi-definite, satisfying the Mercer's conditions [161], then there will exist such higher dimensional feature space and thus the mapping ϕ . That will guarantee that k can be written as a dot product, being a legitimise kernel function.

Some popular kernels used in practice are:

- **Polynomial:** $k(\mathbf{x}_i, \mathbf{x}_j) = (1 + \langle \mathbf{x}_i, \mathbf{x}_j \rangle)^p$
- **Radial Basis Function (RBF):** $k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\frac{1}{2\sigma^2} \|\mathbf{x}_i - \mathbf{x}_j\|_2^2)$

Recursive Feature Elimination Support Vector Machines

SVM has proven state of the art performance in a wide range of applications [14, 80] in particular with problems of enormous dimension, scaling very well with the number of examples. As we have seen previously, for a given set of data examples, SVM aims to find the hyperplane with the longest distance to the nearest training data points of any class. The solution of the optimization problem Eq. 2.28 provides the support vectors, i.e., points lying on the max-margin hyperplane of separation between classes and the coefficients $\mathbf{w}$ of such a hyperplane. The coefficients $\mathbf{w}$, are expressed as an affine combination of the support vectors α_i , the class labels y_i and the training examples $\mathbf{x}_i$. These coefficients can be interpreted as the strength or contribution of each feature to the decision of the hyperplane. As a consequence, the square value of each coefficient (or weight) can be used as a score to rank features from least to the most important for the selection process [83].

SVM has been extended as a form of *embedded* feature selection algorithm. The Recursive feature elimination SVM (RFE-SVM) [83] is an iterative algorithm integrating a ranking criterion for eliminating features in a backward fashion. Starting with the entire set of features, a linear SVM is estimated using the training set, and their features are ranked according to the weights,

i.e., the coefficients $\mathbf{w}$ retrieved by the algorithm in Eq. 2.28. Consequently, the least important features are removed and the remaining ones are used to train a new model, repeating the process until reaching a desired minimal subset of features.

The RFE-SVM algorithm has a set of internal parameters influencing the computational complexity and the accuracy of the method. The fraction features to remove at each step of RFE (also called step size) is critical for the running time. Dropping one feature at a time allows a finer selection but with a prohibitive computational cost. The iterative process will continue until it reaches a minimum number of features, as the stopping criterion for the algorithm.

Multi-class SVM

SVMs are inherently binary classifiers. A traditional way to do multi-class classification with SVM relies on a popular heuristic known as one-vs-all (OvA), or one-vs-rest (OvR) strategy [16]. It consists of fitting one classifier per class, fitting each class against all other classes within each classifier. Making a prediction consists of applying all classifiers to the unseen example and predicting the label for which the corresponding classifier reports the highest score. An advantage of this approach is that as each class is represented by only one classifier, it is possible to gain knowledge about the class by inspecting the individual classifier.

In Chapter 4 we will present some multi-class SVM classification of real-life networks.

Random Forest

Random Forest (RF) [25] is a popular algorithm based on aggregation principles for classification and regression problems. RF operates by constructing and training an ensemble of decision trees over a subset of features. Those decision trees vote on how to classify an unseen data example, and the RF uses these votes to choose the best prediction.

As many complex models with low bias compose the random forest algorithm, it is prone to over-fitting. In order to overcome this limitation, RF uses a general bagging technique, by fitting decision trees on random subsamplings (with replacement) of the training set. The decision for the final prediction for a testing data point is taken by aggregating the output of individual trees,

using, for instance, the majority vote criterion in classification tasks. This procedure leads to better model performance because it decreases the variance of the model without increasing the bias. The number of samples (or trees), is a free parameter which can be tuned in practice using cross-validation.

In Chapter 4, we will use RF to perform supervised graph classification in chemical graphs and social networks.

Unsupervised learning

Unsupervised learning algorithms is about inferring patterns from data without having any prior knowledge about the target label for data examples. Unlike supervised learning, unsupervised methods cannot be applied directly on regression or classification problems because we ignore the real output of that we want to predict, i.e. a label associated to a data example, making impossible to train an algorithm with a supervised objective.

Some examples of unsupervised learning techniques we used in this thesis are:

- **Clustering.** It consists in partitioning a collection of data points $\{\mathbf{x}_i, 1 \dots, m\}$ into K meaningful groups $C_1, C_2, \dots, C_K$ known as clusters. Points within a cluster are more similar between them than those in different groups. A popular clustering algorithm is K -means.
- **Anomaly detection.** It is about uncovering unusual patterns on the dataset, i.e., outlier points. It is commonly used in fraud detection, outlier detection, and identifying fake accounts in social media.
- **Dimensionality reduction.** This technique aims at representing high-dimensional data in low-dimensional spaces in order to facilitate their visual interpretation.
- **Representation learning.** Also known as automatic feature learning, aims to automatically discover representations from raw data, needed for performing downstream machine learning tasks, e.g., classification, clustering, or visualization, i.e., graph embeddings.

In the following, an unsupervised neural network approach is presented, as used in later chapters.

Autoencoder (AE)

The auto-encoder (AE) [69,183] is one of the most popular unsupervised neural network approaches. It has been broadly used as a performant mechanism to pre-train neural networks and general-purpose feature learning [37]. The AE is a neural network that is trained to attempt to copy the input data to the output. It is composed by two parts: an encoder function $\mathbf{y} = f_{\theta}(\mathbf{x})$ and a decoder that reconstructs the input $\mathbf{z} = g_{\theta'}(\mathbf{y})$. Its architecture is presented in figure 2.4. Allowing the AE to reconstruct solely its input is not especially useful. Instead, AEs are restricted in different ways to force a partial reconstruction of the input so that the model will prioritize the main features of the input data obtaining useful properties from it. For instance, one way to obtain useful features from the AE is constraining the $\mathbf{y}$ to have a smaller dimension than $\mathbf{x}$. By doing so, the AE will compress the input data, capturing the main factors of variability and producing useful representations of the input.

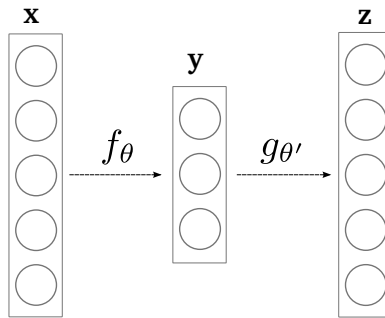


Figure 2.4: Architecture of the undercomplete autoencoder. The encoder and decoder functions are typically non-linear functions, with parameters estimated through stochastic gradient descent.

More formally, given a set of data examples $\mathcal{D} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(m)}\}$, the purpose of the auto-encoder is to learn a non-linear mapping which encodes an input example $\mathbf{x} \in \mathbb{R}^n$ in a smaller dimensional latent vector $\mathbf{y} \in \mathbb{R}^d$ with $n \gg d$. The encoding mapping has the form of $f_{\theta}(\mathbf{x}) = s(\mathbf{W}\mathbf{x} + \mathbf{b}) = \mathbf{y}$, generally through a non-linear function s such as sigmoid or $\tanh$ applied entrywise on the vector $\mathbf{W}\mathbf{x} + \mathbf{b}$. A reverse mapping of f is used to reconstruct the input from the feature space: $g_{\theta'}(\mathbf{y}) = s(\mathbf{W}'\mathbf{y} + \mathbf{b}') = \mathbf{z}$. The parameters $\theta = \{\mathbf{W}, \mathbf{b}\}$ and $\theta' = \{\mathbf{W}', \mathbf{b}'\}$ are estimated by minimizing the average reconstruction error

over the training set:

$$\theta^*, \theta'^* = \arg \min_{\theta, \theta'} \frac{1}{m} \sum_{i=1}^m \|\mathbf{x}^{(i)} - \mathbf{z}^{(i)}\|_2^2 \quad (2.34)$$

Note that when s is the trivial identity, the solution is equivalent to the classical PCA (principal component analysis) with the number of hidden units as the principal components. One can, therefore, see autoencoders as a non-linear extension of PCA.

3

Anomaly detection on attributed networks

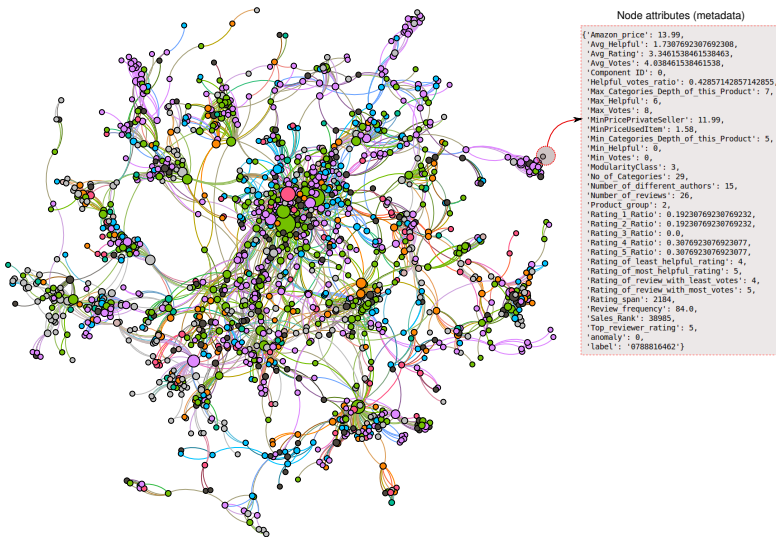


Figure 3.1: Anomaly detection on attributed networks. The plot shows the Amazon co-purchase network of books [131]. Nodes have multiple attributes. Colors represent the selling price. The challenge relies on spotting anomalous books whose attributes are in disagreement with other books in different neighbourhood scales.

In this chapter, we introduce a graph mining algorithm (*MADAN*), to perform **Multi-scale Anomaly Detection on Attributed Networks**. Many social and economic systems can be represented as attributed networks encoding the relations between entities who are themselves described by different node attributes, see Figure 3.1. The task of finding anomalies in these systems is essential for detecting abuses such as credit card frauds, web spams, or network intrusions.

The main contribution in this chapter is the proposition of a novel anomaly detection framework based on the dynamics of a diffusive process on an attributed network. Our approach offers a principled way to simultaneously uncover

outlier nodes, as nodes with a high concentration in the diffusive process, and the contexts in which each anomaly is relevant, as clusters of nodes retaining the diffusive flow for a given time.

Our framework exploits the link between multi-scale community detection in networks, Section 2.1.6, based on the diffusion of random walks, and the low-pass filtering of graph signals with heat kernels. The performance of our method is assessed experimentally on synthetic and real-life attributed networks, being particularly efficient computationally when the diffusive process is approximated with Chebychev polynomials.

The results of this chapter are in collaboration with Alexandre Bovet and Jean-Charles Delvenne.

Introduction

Anomaly detection is an important problem in data mining with multiple applications in diverse domains. Examples include credit card fraud detection [11], intrusion detection for individual and network systems [53, 198], identifying abnormal users in communication networks [187], webspam detection [35] and so on. Anomalous data can be understood as unusual objects with patterns or behaviors that deviate significantly from any background property. Moreover, many real-world complex systems are often well modeled by a network consisting of nodes with a certain number of attributes and edges representing the relations between nodes. For instance, in a social network, where nodes usually represent persons, and edges represent social relations, node attributes could be demographics information. In a co-purchase network, nodes represent items and might contain information about them as attributes, while an edge is present between two items when the same person has bought them.

Because the mechanisms generating anomalies in real-world networks are usually unknown, defining ground truth anomalies is problematic and measuring the degree for which a node is anomalous is a challenging problem. For this reason, anomalies must be defined in opposition to a background of ‘normal’ nodes, i.e., the *context* relevant for an anomaly. For instance, social networks are known to have highly homophilic communities, i.e., people tend to create social ties with similar persons, in terms of age, ethnicity, etc.

In a social network, considering the income as a social attribute, people with similar income are more likely to be within the same community. Therefore, someone with a high income having a friendship circle with low-income people

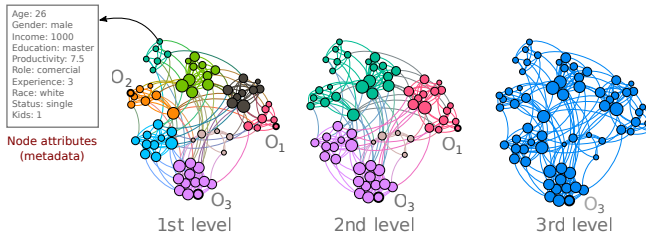


Figure 3.2: A toy example of work relation network. In this work, multi-scale refers to the analysis of the network at different time scales in order to uncover robust partitions. Nodes have attributes describing individual features. Node attributes define structural clusters in multiple scales. At the *1st scale* outlier nodes (O_1, O_2, O_3) lie within a local context, i.e, offices. In a *2nd scale*, departments emerge as new contexts where O_2 is not defined. Finally, at a larger scale O_3 remains as a global anomaly in context of the whole company.

would be considered as an outlier node for a surrounding context of a low-income group of people.

Anomaly detection is therefore naturally linked to the problem of community detection, which consists of finding clusters of nodes that are densely linked together compared to the rest of the network [51]. By performing community detection on a network where related nodes are close to each other, and edge weights express the similarity of nodes' attributes, we discover all the relevant contexts for anomalous nodes.

Moreover, real systems that can be represented as a network are characterized by different levels of communities, depending on the scale at which the network is investigated. Thus, in this work we call multi-scale the fact that we perform our analysis across all time scales of the graph in order to uncover robust partitions at each scale. For instance, as is shown in Figure 3.2, colleagues in an office may represent communities at a small scale in a work relation network. At a larger scale, one may discover departments as structural communities, and at even larger scale companies may form communities. Additionally, anomalies might be only relevant for a surrounding context of a given size but may disappear when a larger context is considered, while other anomalies may be relevant at all scales. Considering the multi-scale nature of real-world systems is therefore critical for the detection of anomalies.

Here we present a novel framework providing a simple physical interpretation for anomalous nodes in attributed networks: they are nodes with a highly

concentrated flow in a diffusive process taking place on the network. Our framework provides a simple and principled mechanism to rank and localize outlier nodes at different scales. See Figure 3.3 for an illustration of this process.

We exploit the equivalence between the filtering of graph signals by a heat kernel operator, originating in the graph signal processing framework [166], and the diffusive process describing a continuous-time random walk, the basis of the concept of Markov stability of a network partition [51, 106].

Although this relationship was introduced in a more general setting in [179] through spectral graph wavelets for performing community detection, here we develop a particular case in a different application.

Applying a heat kernel operator on a unit impulse graph signal, we gradually smooth the signal according to the network structure and the similarity between nodes attributes, identifying outliers as nodes being hardly smoothed for any time scale. From a dynamics perspective, the values of localized heat filters can be seen as the diffusion of a unit of heat from vertex to its surrounding vertices over a fixed time. By considering edge weights that depend on the similarity of node attributes such that heat diffuses faster between nodes with similar attributes, regions having similar node attribute values correspond to regions with a low concentration of heat. In contrast, outliers nodes correspond to nodes where the heat remains highly concentrated.

Simultaneously, the corresponding context is provided by the Markov stability framework that finds the partition in clusters of the network that maximizes the stability of the heat flow, i.e., such that the flow is more likely to remain in the same clusters after a given time [51]. As the anomaly detection and the context detection are both based on the computation of the same diffusive process, our method discovers the contexts of anomalies without additional computational cost.

Tests on real-world benchmarks show that our method allows not only to recover and rank the so-called ground truth anomalies but also to discover new anomalies jointly with their contexts. In addition, empirical experiments show that our method is scalable for large networks, based on Chebychev approximations of the exponential of the graph Laplacian. Our work shows that anomaly detection on networks requires a principled investigation considering simultaneously the anomalies and the context in which they are relevant.

This chapter is organized as follows: Section 3.5 reviews related approaches in the literature. Section 3.2 introduces filtering graph signals and clustering attributed nodes, finalizing with the proposed anomaly detection technique.

In Section 3.3 we present our experimental setting, with validation in diverse datasets as well as computational aspects of the method. A case of study is presented in detail in Section 3.4, finalizing in Section 3.6 with conclusions and perspectives.

Filtering graph signals and clustering attributed networks

Graph signals on attributed networks

Let us represent our systems with a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$ composed by a set of vertices $\mathcal{V}$ with $|\mathcal{V}| = N$ nodes, a set of edges $\mathcal{E}$ and a weighted adjacency matrix $\mathbf{W}$. For simplicity, we will restrict this work to undirected and connected graphs. Nevertheless, this definitions are also extensible to the directed and disconnected cases [39]. A graph signal is a function $f : V \rightarrow \mathbb{R}^+$ defined on the vertices of a graph, i.e. node attributes, and can be represented as a vector $\mathbf{f} \in \mathbb{R}^N$, with the i -th element of $\mathbf{f}$ representing an attribute value at vertex i . Considering multidimensional node attributes, we associate to each vertex $u \in V$ a d -dimensional vector $\mathbf{f}(u) = \langle \mathbf{f}_1(u), \dots, \mathbf{f}_d(u) \rangle$ where $\mathbf{f}_k(u) \in \mathbb{R}^+$ represents the k -th attribute of the vertex u .

To each edge of the graph, we associate a weight expressing the similarity of the nodes' attributes linked by the edge. Here we use the Gaussian kernel weighting function:

$$W_{u,v} = \begin{cases} \exp\left(-\frac{\|\mathbf{f}(u) - \mathbf{f}(v)\|^2}{2\sigma^2}\right) & \text{if } (u, v) \in \mathcal{E} \\ 0 & \text{otherwise} \end{cases} \quad (3.1)$$

where σ is a scaling parameter. As the graph is undirected, $\mathbf{W}$ is a symmetric matrix. Let $\mathbf{d}_u = \sum_{v \in N_u} W_{u,v}$ be the strength of the node u , i.e. the sum of the edge weights connecting u with its direct neighborhood N_u . The combinatorial matrix is defined as $\mathcal{L} = \mathbf{D} - \mathbf{W}$, where $\mathbf{D}$ is the diagonal matrix with $\mathbf{d}$ as diagonal i.e. the vector of all node's strengths.

For simplicity in the presentation of the subsequent theory, we drop the subscript k let us denote an arbitrary graph signal by $\mathbf{f} \in \mathbb{R}^N$. For any given graph signal, the Laplacian $\mathcal{L}$ acts as a difference operator such that the expression:

$$\langle \mathbf{f}, \mathcal{L}\mathbf{f} \rangle = \sum_{(u,v) \in E} W_{u,v} (\mathbf{f}(u) - \mathbf{f}(v))^2 \geq 0 \quad (3.2)$$

provides a global measure of the variation (smoothness) of $\mathbf{f}$ along the graph edges [166]. Because $\mathcal{L}$ is real-symmetric, it has non-negative eigenvalues $\lambda_0 < \lambda_1, \dots \leq \lambda_{N-1}$ and a set of associated orthonormal eigenvectors $\{\mathbf{u}_l\}_{l=0,\dots,N-1}$. When the graph is connected there exist one zero eigenvalue whose algebraic multiplicity equal to one.

It turns out that the smoothness of a Laplacian eigenvector $\mathbf{u}_l$ is equal to its corresponding eigenvalue, i.e. $\langle \mathbf{u}_l, \mathcal{L}\mathbf{u}_l \rangle = \lambda_l$. Eigenvectors associated with small eigenvalues are therefore smoother than eigenvectors associated with large eigenvalues which are more likely to have dissimilar values on vertices connected by an edge with a large weight. Thus, Laplacian eigenvectors can be interpreted as Fourier modes of the graph with their eigenvalues as corresponding ‘frequencies.’ Denoting $\mathbf{U} = [\mathbf{u}_0, \dots, \mathbf{u}_{N-1}]$, the Laplacian can be written as:

$$\mathcal{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T \quad (3.3)$$

where $\mathbf{U}^T$ denotes the transpose of $\mathbf{U}$ and $\mathbf{\Lambda}$ the diagonal matrix with the eigenvalues of $\mathcal{L}$ in its diagonal. For any graph signal $\mathbf{f} \in \mathbb{R}^N$, its graph Fourier transform (GFT) is defined as $\hat{\mathbf{f}} = \mathbf{U}^T \mathbf{f}$ and its inverse graph Fourier transform (IGFT) as $\mathbf{f} = \mathbf{U} \hat{\mathbf{f}}$ (see Ref. [167] for complete discussion about graph Fourier transformations).

Filtering of graph signals

Graph Fourier transform plays a central role in graph signal processing allowing a natural extension of filtering operations. In the classical signal processing setting [170] the filtering of a signal is defined as a convolution of a signal with a filter, which is equivalent to a point-wise product in the spectral domain. In the graph domain, a graph filter (or kernel) is defined in terms of a continuous kernel $g : \mathbb{R}^+ \rightarrow \mathbb{R}$ evaluated on the Fourier frequencies of the graph. Thus, filtering a graph signal $\mathbf{f} \in \mathbb{R}^N$ with a kernel g reads [166]:

$$g(\mathcal{L})\mathbf{f} = \mathbf{U}g(\mathbf{\Lambda})\mathbf{U}^T \mathbf{f} \quad (3.4)$$

As the filtering operation relies on the diagonalization of the Laplacian of the graph, its complexity is $O(N^3)$ making it impractical for large graphs. However, in our applications, we used existing faster algorithms based on Chebyshev polynomials to approximate the kernel computations, whose complexity scales linearly with the number of edges of the graph [168]. More details about the computational issues can be found in Section 3.6.

Clustering attributed networks

The problem of partitioning a network in clusters that reveal its modular structure can be solved by considering the trajectories of random walks on the network and by finding the clusters in which random walkers remain for a long time [51, 106, 157]. This framework called the Markov stability framework [51], allows uncovering the multi-scale structure of networks by studying the diffusive process describing the random walks at different times. For a more detailed introduction to this framework, we refer the reader to the Section 2.1.6.

Here, we consider the undirected weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$ with N nodes defined in section 3.2.1 where nodes attributes are incorporated in the computation of the edge weights. In this case, the Markov stability framework allows finding clusters of nodes that are densely connected in terms of network distance and attribute similarity, simultaneously.

We consider a continuous time random walk on this network, i.e., the heat diffusion Eq. 2.14, for the walker density vector $\mathbf{p}(t)$ at time t :

$$\frac{d\mathbf{p}(t)}{dt} \langle \mathbf{d} \rangle = \mathbf{p}(t)(-\mathbf{D} + \mathbf{W}) = -\frac{\mathbf{p}(t)}{\langle \mathbf{d} \rangle} \mathcal{L} \quad (3.5)$$

where $\mathcal{L}$ is the combinatorial Laplacian matrix of the graph. This equation implies that the walker transition rate at a node u is inversely proportional to the node strength $\mathbf{d}(u) = \sum_{v \in N_u} W_{u,v}$ [106, 126, 143]. The stationary solution of Eq. 3.5 is the uniform vector:

$$\boldsymbol{\pi} = \mathbf{1}^T / N \quad (3.6)$$

where $\mathbf{1}$ is a $N \times 1$ vector of ones [126].

Each partition of the graph into K clusters is encoded in a $N \times K$ indicator matrix $\mathbf{H}$ with element $h_{i,j} = 1$ when the node i belongs to the cluster j and $h_{i,j} = 0$ otherwise. Denoting $\boldsymbol{\Pi} = \text{diag}(\boldsymbol{\pi})$, the element (i,j) of the so called clustered autocovariance matrix, introduced previously in Eq. 2.1.6 is:

$$\mathbf{R}(t; \mathbf{H}) = \mathbf{H}^T (\boldsymbol{\Pi} e^{-\frac{t}{\langle \mathbf{d} \rangle} \mathcal{L}} - \boldsymbol{\pi} \boldsymbol{\pi}^T) \mathbf{H} \quad (3.7)$$

which encodes the probability for a random walker to go from cluster i to cluster j during a time interval t , minus the same probability for two independent random walkers, evaluated at stationarity. The continuous-time combinatorial Markov stability [106] was presented in Eq. 2.19 as

$$r(t; \mathbf{H}) = \text{Tr} [\mathbf{R}(t; \mathbf{H})], \quad (3.8)$$

and measures the quality of a network partition as the tendency of the diffusive flow to remain within its initial cluster after a time t .

By finding the partition that maximizes $r(t; \mathbf{H})$ for increasing time scales t , this framework allows to reveal the finer and coarser clustering of the network which allows recovering the hierarchical configuration of the network structure and nodes attributes. For more details about the Markov stability framework, we refer the reader to Refs. [51, 106, 157].

Given that the maximization of Eq. 3.8 is an NP-hard problem [24] there is no guarantee of optimality. However, it can be written as a modularity maximization problem for which the Louvain algorithm [18] is known to provide good results in practice, see Section 2.1.6.

In order to uncover the relevant time scales of the network, we perform a systematic scan of relevant partitions across times. That is, we maximize Eq. 3.8 across Markov times, obtaining a sequence of optimized partitions. Besides, to assess the robustness of retrieved partitions at a given time scale, we proceed as was presented in Section 2.1.6. That is, we run the Louvain algorithm 100 times with different random initializations and compute the variation of information [127] among the ensemble of retrieved partitions for a given time, i.e., $VI(P_1, P_2)(t)$.

To simplify the notation, we denote $VI(P_1, P_2)(t)$ as $VI(t)$. That is, the normalized variation of information between partitions P_1 and P_2 retrieved at time scale t . To find the range of time scales relevant for a given partition, we also compute the variation of information between the ensemble of optimal partitions found between each pair of times $VI(t, t')$, as shown in Eq. 2.24. Thus, relevant partitions $\mathbf{H}$ are finally identified by Markov times t, t' where $VI(t)$ shows a dip and $VI(t, t')$ shows a continuous plateau [106]. See Figure 3.3 for an example.

Anomaly detection

Unlike in standard signal processing where filters are time-translation invariant [170], the effect of filtering a Kronecker delta signal e.g. $\delta_a(k) = 1$ if $k = a$ and 0 otherwise, depends on the node where the signal is concentrated, because it depends on the local structure around the node [167].

The effect of a low-pass filter such as the heat kernel, known in the literature as smoothing [166, 197], will inform us on how well the node is connected with its neighborhood. If the node is strongly connected (edges of high weight) with its neighbors, themselves strongly connected with their neighbors, etc., then the

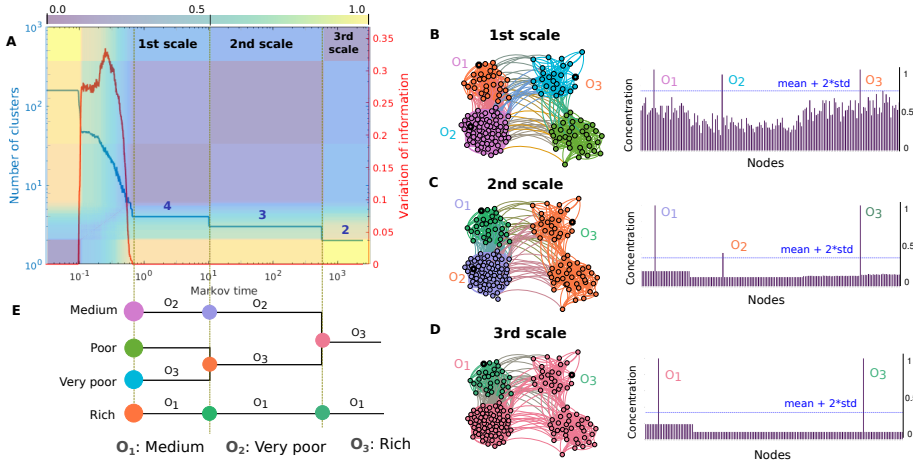


Figure 3.3: Synthetic social network of 160 nodes with a planted partition of four communities. Each node (representing a person) has a scalar attribute representing the income of a person, partitioning the network in four clusters labeled as rich, medium, poor, and very poor. Three outlier nodes are injected: O_1 is a medium income person within the rich cluster, O_2 a very poor person within the cluster of medium income people and O_3 is a rich person in the very poor cluster. (A) We perform a scanning of relevant contexts along the attribute subspace: for each time t we find the number of clusters (blue curve) and compute the average variation of information ($VI(t)$) of the retrieved partitions (red curve). The $VI(t)$ is obtained from 100 runs of the Louvain algorithm. The background represents the $VI(t, t')$ between optimized partitions across times. As can be seen in (A), starting at $t \approx 0.65$, the partitions of 4, 3 and 2 clusters are found as persistent according with the low $VI(t)$. Subsequently, we compute the concentration over all nodes of the graph (bar plots) with Eq. (3.9). (B) At the 1st scale, the concentration of nodes at $t = 1.5$ reveals three local outliers with their context given by the colored clusters. (C) At the 2nd scale, at $t = 10.5$ the context of O_3 enlarges whereas the concentration of O_2 decreases. (D) At the 3rd scale given by $t = 578$, O_2 is not an outlier and the context for O_3 enlarges more. (E) The dendrogram shows the hierarchy of partitions and the outlier nodes at each scale.

smoothed δ_a will expand on a wide neighborhood. If the node is completely isolated, δ_a will remain unalterable, because no diffusion is possible towards neighbors. If a node is poorly connected with its neighbors, then δ_a will be somewhat smoothed. In other terms, the concentration of the δ_a through a heat kernel filtering will be indicative of this connectivity.

Concentration is measured with the L_2 -norm [150] of the filtered δ_a signal with a heat kernel at time $\frac{t}{\langle d \rangle}$ as:

$$c_a(t) = \|e^{-\frac{t}{\langle d \rangle} \mathcal{L}} \delta_a\|_2. \quad (3.9)$$

This measure is meaningful because the filter preserves the sum (or mean) of the signal, and therefore the Kronecker delta has a maximum norm of one while the completely smoothed signal has norm $1/N$, which is the smallest

possible norm for a signal of sum one spread over N nodes.

It is worth to mention that filtering a δ_a signal with a heat kernel at a scale t can be interpreted as a diffusion of a unit of heat energy from a central vertex a to the surrounding vertices over a fixed time t . Indeed, the heat equation is the dual of Eq. 3.5 and, in our case of an undirected and connected network, the heat kernel is the propagator of the diffusion equation Eq. 3.5.

Heat diffuses with a rate given by the edge weights such that heat spreads faster between nodes with similar attributes making the localized filter kernel less concentrated. In contrast, heat remains more concentrated on nodes poorly connected in terms of edge weights slowing down the diffusive flow across the graph nodes. Note that the sigma parameter in Eq. 3.1 controls the rate of diffusion across edges. Thus, increasing sigma will accelerate the diffusion rate globally, and the context at a given scale will be coarser than it would be for smaller sigma.

We use the concentration as a way to quantify the degree of deviation of a given node with respect to a context at a time scale and provides scoring for ranking potential outliers. Nodes having a high concentration value are considered as more abnormal than nodes with a lower score. However, a practical criterion is needed to characterize anomalous nodes.

In this work, we use a standard thresholding rule used in general outlier detection methods [93], which works well in our experiments. However, we stress that a more efficient criterion for selecting a proper threshold can be developed from the concentration distribution.

Let $c(t) = [c_1(t), \dots, c_N(t)]$ be the overall graph concentration. A node a is considered as anomalous at a time scale t if the following thresholding condition holds:

$$c_a(t) \geq \bar{c}(t) + 2s(c(t)) \quad (3.10)$$

with $\bar{c}(t)$ as the average concentration and $s(c(t))$ its standard deviation.

Context for anomalous nodes

The interpretation of the filtering of a Kronecker delta with a heat kernel as a diffusive process on the network allows us to link anomaly detection with the framework of Markov stability (Eq. 3.8). As anomaly detection is naturally linked with the problem of clustering of attributed nodes on networks, the context of an anomalous node is provided simultaneously by the best partition uncovered by Markov stability evaluated with the heat diffusion at the same time than the time scale used to evaluate the concentration Eq. 3.9. For small

time scales, outlier nodes will exist only on narrow contexts characterized by finer partitions of the network in the node attributes subspace. For larger times, coarser partitions emerge as larger contexts for outlier nodes persisting over many structural communities. The effect of the Markov dynamics as a zooming lens provides a means to uncover anomalous nodes with respect to a context organized hierarchically across different scales. Figure 3.3 illustrates an example of our approach.

Our multi-scale anomaly detection method can be summarized as follows:

1. We perform scanning of relevant partitions computing $VI(t)$ and $VI(t')$ for increasing times.
2. Choose a scale t relevant to the desired partition (finer, coarser)
3. Compute the Fourier basis with a scaled heat kernel at time t localized at each vertex $a \in V$, e.g., filtering a Kronecker delta signal centered on a with a heat kernel evaluated at scale t (Eq. 3.4).
4. Compute the node concentration for each node (Eq. 3.9)
5. The context in which a node is considered as an outlier will be given by the optimal partition retrieved by the optimization of the Markov stability (Eq. 3.8) at the same scale than the one used to evaluate the Fourier basis.

Experiments

In this section, we evaluate the performance of our method in spotting anomalous nodes in synthetic and real-life attributed networks. We compare our method, *Multi-scale Anomaly Detection in Attributed Networks* (MADAN), against representative methods of the literature.

Synthetic dataset

We generate a series of undirected synthetic networks with ground truth community structure, adding node attributes, and injecting node anomalies in different hierarchies.

In order to simulate real-life networks, we generate power-law networks from the Lanchinetti-Fortunato-Radicchi (LFR) benchmark [61]. This algorithm generates networks with diverse community structures, community sizes, and

power-law degree distributions as many real-world networks. Networks with $N = 1000$ nodes are generated with the mixing parameter $\mu = 0.1$ generating partitions of well-defined communities containing between 50 to 200 nodes. The average node degree is ranging from 10 to 100. The sizes of the communities are taken from a power-law distribution with exponent 1. Similarly, the degree distribution for the nodes is drawn from a power-law distribution with an exponent equal to 2.

Nodes attributes are generated as follows: to each node, we associate a vector of attributes of dimension $d = 20$. To make the problem more challenging, we introduce heterogeneity in the node attribute values, so that within each community $C \in \{0, 1, 2, \dots, c_{max}\}$ nodes attributes are sampled from different probability distributions.

For any attribute k of a node belonging to a community C , its values are drawn from a normal distribution with mean C and standard deviation σ , a uniform distribution in the interval $[C - \sigma, C + \sigma]$ and a logistic distribution with mean C and scale σ . As nodes attributed within the same cluster should agree in their values, we chose $\sigma = 0.1$ for all distributions. Besides, we force some communities to have similar node attributes, i.e., drawn from the same distributions, in order to introduce attribute hierarchies. Thus, the networks contain structural communities, given by the ground truth community structure, and attribute communities on top grouping some communities into the same attribute schema.

To generate anomalous nodes, a varying percentage of randomly chosen nodes ($\{1\%, 5\%, 10\%, 15\%, 20\%, 25\%, 30\%\}$) is perturbed by replacing 30% of their attributes with attributes from a different, randomly chosen cluster. The performance of anomaly detection is measured over 50 randomly generated networks, reporting its average and standard deviation.

Real life datasets

We use three attributed networks labeled with the so-called ground truth anomalous nodes used as benchmark datasets in the literature. We summarize their general properties in Table 3.1. The Disney and Books datasets [131] are co-purchase networks extracted from Amazon. They contain 28 attributes per node describing properties about online items, i.e. rating, selling price, number of views, etc. The ground truth anomalies for Disney DVDs movies were tagged manually by high school students in Germany. A partition of the network was obtained after applying a modularity maximization algorithm.

Table 3.1: Real life datasets

DATASET	# nodes	# edges	# node attributes	# anomalies
Disney	124	334	28	6
Books	1418	3695	28	28
Enron	13533	176987	20	5

Subsequently, students were asked to tagged anomalous nodes regarding a high deviation of the attribute values of the nodes within each cluster. Outliers nodes were defined as those whose were tagged as anomalous by at least 50% of the students [131].

Before 2012, Amazon users were allowed to tag online products in order to improve the organization of products and generate user recommendations. In particular, several tags were used for customers to indicate novelty, suspicious, or rare products. The tag *amazonfail* was used during many years for users expressing their disagreement with the sales rank of an item. Thus, this tag was used as ground truth for anomaly detection in the Books dataset [131].

Enron [128] is a communication network with edges indicating email transmission between people. Each node contains 20 attributes describing metadata of the message i.e., content length, number of recipients, etc. This dataset has been extensively used as a benchmark for spam detection [128]. Spammers were labeled as ground truth for anomaly detection [131].

Experimental settings

Following the evaluation setting of [2,114,131], we assess the anomaly detection performance with two well known metrics for evaluation of anomaly detection systems [2]: the area under the receiver operating characteristic curve (AUC-ROC) and the area under the precision/recall curve (AUC-PR). The former is created by plotting the true positive rate (tpr) against the false positive rate (fpr) at various threshold settings. That is, for a given threshold $\alpha \in [0, 1]$, we compute the tpr and the fpr determined by the thresholding rule $c_a(t) \geq \alpha$, for all nodes of the graph. The tpr is defined as the detection rate, i.e., the rate of actual anomalous nodes correctly identified as anomalous, whereas the fpr is the false alarm rate, i.e., rate of normal nodes identified as anomalous. Thus, we report the area under that curve as a function of perturbed nodes (anomalous nodes). The second metric quantifies the trade-off between the

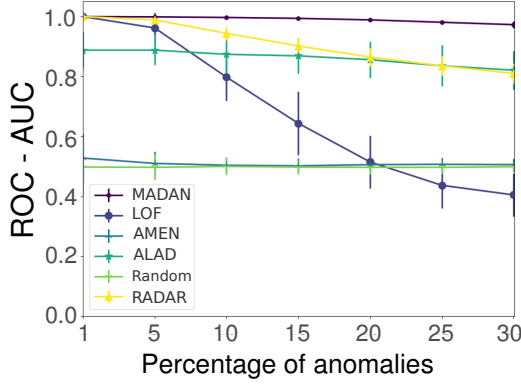


Figure 3.4: Synthetic experiments. Performance on anomaly detection when increasing levels of perturbed nodes are injected. Colors corresponds to different baseline techniques, including our MADAN approach. We report the mean **ROC-AUC** and standard deviation (vertical lines) over 50 LFR random networks.

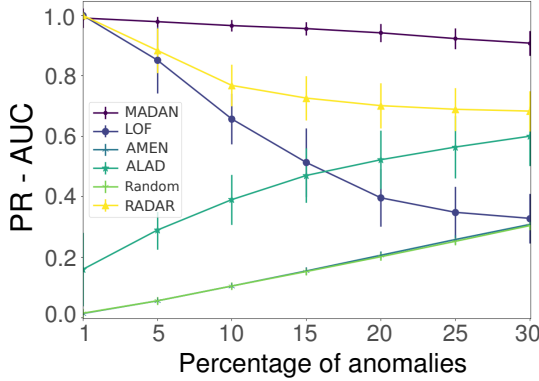


Figure 3.5: Synthetic experiments. Performance on anomaly detection when increasing levels of perturbed nodes are injected. Colors corresponds to different baseline techniques, including our MADAN approach. We report the mean **PR-AUC** and standard deviation (vertical lines) over 50 LFR random networks.

precision, i.e., how good the model is at predicting the anomalies, and the recall, defined as the ratio between true positive over true positives plus false negatives. That is, for a given threshold $\alpha \in [0, 1]$, we compute the precision and the recall determined by our classification rule $c_a(t) \geq \alpha$, for all nodes of the graph. Thus, we report the area under that curve as a function of perturbed nodes (anomalous nodes). In both cases, the more the area under the curve (AUC) approaches to 1, the better the performance of the method.

We compare our MADAN approach with four baseline methods of the lit-

erature: Local Outlier Factor (LOF) [26], Accelerated Local Anomaly Detector (ALAD) [117], Anomaly Mining of Entity Neighborhoods (AMEN) [148], Residual Analysis for Anomaly Detection (RADAR) [114] and a simple Random classifier (Random) for anomaly detection, i.e. assigning random ranking scores to each node.

Each method will output either an abnormality score or a ranking of nodes, used to quantify the degree of the anomaly. We compute the ROC and PR curves and report their AUC score.

Results

Results on synthetic attributed networks are shown in Figure 3.4 and Figure 3.5. As can be seen, our MADAN approach outperforms the baseline methods in all cases. Indeed other methods are capable of spotting local outliers when attributes and community structure are correlated, whereas our approach can deal with the hierarchical composition of attributes across communities. Injecting a little number of anomalies, i.e., 1% of perturbed nodes, MADAN is comparable with RADAR and LOF. However, when the number of perturbed nodes increases, LOF performance falls because it does not exploit the graph structure. It can also be seen that AMEN performance is precarious comparable to a random classifier because this method was designed to detect abnormal nodes on ego-networks discovering anomalous neighborhoods. MADAN also shows the lowest variance across identifying outliers across different random networks.

Results on real life datasets are shown in Table 3.2. We see that our approach (MADAN) achieves the best AUC-ROC performance compared to other baseline methods on the Disney and Books networks, while Consu+DisOut (C+D) shows the best AUC on the Enron dataset. The advantage of our method that takes into account the structure of the network at different scales, allows it to perform better, or at least with comparable AUC than the baseline methods, in particular compared to LOF, which ignores the underlying structure of the network, and AMEN, which focuses only on anomalous neighbors.

It is worth to mention that, when assessing the performance of anomaly detection algorithms on empirical datasets, one should keep in mind that the set of manually annotated, so-called, ground truth anomalies does not represent an objective truth as different annotators may label different nodes as abnormal, choosing different criteria to make their decisions. We believe that our method, allowing to measure the degree of anomalousness of nodes as a function of

Table 3.2: Results real life datasets (AUC-ROC)

DATASET	LOF	C+D	RADAR	ALAD	AMEN	MADAN
Disney	0.61	0.81	0.87	0.70	0.52	0.93
Books	0.49	0.60	0.58	0.43	0.47	0.68
Enron	0.44	0.74	0.65	0.72	0.47	0.66

its attributed context, offers a framework to explore dataset anomalies in a principled way. We present an example of such an anomaly exploration in the next section.

Computational Issues

The computational bottleneck of our approach resides in the impractical diagonalization of the graph Laplacian for large graphs (more than 8000 nodes), used to compute both the node’s concentration and the Markov stability. A general way to approximate kernel evaluations of general filters on graphs is defined in [87] using an efficient formulation in terms of truncated Chebychev polynomials. On the other hand, Markov stability is computed in [106] by approximating the exponential of the Laplacian with Padé approximations. We observe that the heat kernel evaluated on the graph Laplacian spectrum at a given time can be calculated efficiently in terms of filtering operators as:

$$e^{-t\mathcal{L}} = [e^{-t\mathcal{L}}\delta_1, \dots, e^{-t\mathcal{L}}\delta_N] \quad (3.11)$$

where each column component is approximated with a Chebychev polynomial as [87], being easily parallelized across columns. The error of approximation is controlled by the degree of the Chebychev polynomial m . The larger the m the better the approximation with a cost of a longer running time. We chose $m = 20$ as has been used before in [87].

We perform an empirical evaluation in the computation of the exponential of the Laplacian for a power-law degree distributed network generated with the Barabási-Albert model [6] increasing the number of nodes N from an initial connected network of 3 nodes, so that produces networks with approximately $3N$ edges. We compare the Fourier decomposition of the graph Laplacian, Padé approximations, Chebychev approximations, and its parallel version.

As can be seen in Figure 3.6, Chebychev approximations are less efficient than the full eigenvector decomposition and Padé approximation of the Lapla-

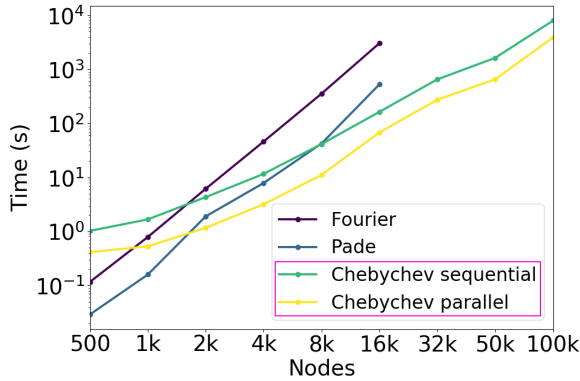


Figure 3.6: Running time for the computation of the exponential of the graph Laplacian for $t = 1$ varying the network size. Our parallel setting run across 8 cores.

cian for small graphs. However, for larger networks, the Chebyshev polynomials are by far more efficient, where its parallel instance is the fastest one.

The parallel setting was performed over 8 cores. All computations were done on a standard computer Intel(R) Core(TM) i7-4790 CPU, 3.60GHzI with 16G of RAM.

Multi-scale anomaly detection: a case of study

We consider the Disney network as a case study. This dataset provides manually labeled ground-truth anomalies [131] and has been largely used as a benchmark for outlier detection in networks. Here, we use *MinPricePrivateSeller* and *Number_of_reviews* as node attributes. We show in Figure 3.7 the Markov stability analysis revealing the existence of several intrinsic scales in the network that form a hierarchy of communities. As the edge weights capture the attribute similarity, the clusters represent the context of both structural closeness and attribute similarity.

The contour plot in Figure 3.7 (A) shows the variation of information $V(t, t')$ between set of optimal partitions at time t and t' and reveals blocks of low variation of information (dark blue blocks) corresponding to the different intrinsic scales of the attribute network. The number of communities at each scales (shown in blue) displays plateaus corresponding to the blocks in $V(t, t')$. The intrinsic scales are also visible as dips in the variation of information among the set of partitions found by iterating the Louvain algorithm at a given time scale, $V(t)$, (red curve).

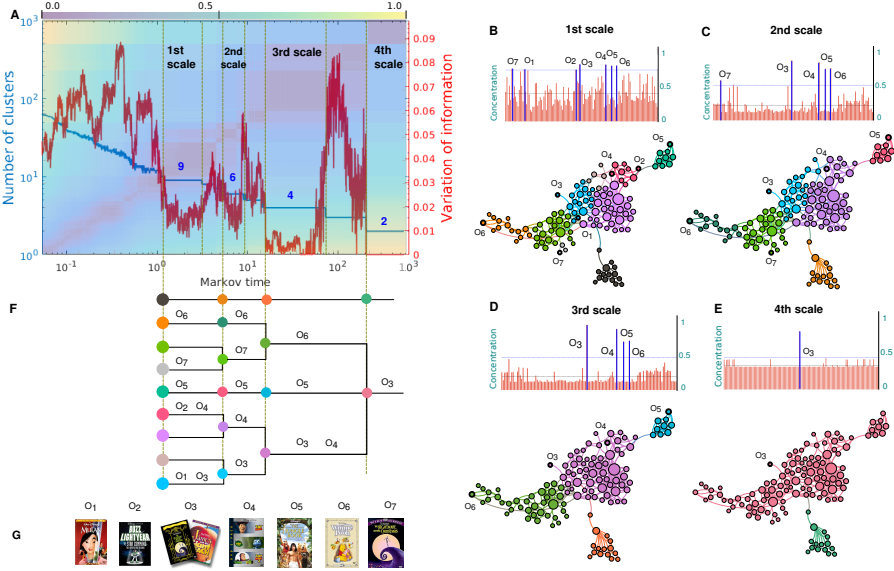


Figure 3.7: (A) Number of clusters (blue curve) found by maximizing Eq. 3.8, at each time step, the variation of information $VI(t)$ (red curve) between the ensemble of optimal partitions at each time and the variation of information $VI(t, t')$ between optimal partitions across times (background contour plot). Relevant partitions are determined by dips of $VI(t)$ and extended plateaus of $V(t, t')$. Visualization of four robust partitions and the node heat concentration (bar plots), indicating outlier nodes (blue bars) when evaluating the node concentration at (B) $t = 1.18$, (C) $t = 7.25$, (D) $t = 14.89$ and (E) $t = 204.8$. In all bar plots, the upper horizontal line (blue) indicates the detection threshold defined in Eq. 3.10. (F) Dendrogram showing the hierarchy of clusters at each time with the contextual outliers. (G) Anomalous Disney DVD movies.

For each intrinsic scales found in Figure 3.7 (A), we display the corresponding optimal partition in the right-hand side of Figure 3.7.

At the same time than the clustering process, the spectral graph basis for the combinatorial Laplacian, i.e., $g(t\mathcal{L}) = e^{-t\mathcal{L}}$, are evaluated at the same scale t than the Markov stability in order to compute the anomaly score as the node heat concentration Eq. 3.9.

As shown in Figure 3.7, (bar plots in A,B,C,D) outlier nodes at a given scale are detected as the ones having a concentration of energy, i.e., more than two standard deviations above the mean concentration Eq. 3.10.

Figure 3.7 (C) shows the partitioning of the network in 9 communities. Node O_5 (*The jungle book*) appears as an anomalous node within its community that corresponds to all the *read along* movies, having both high selling price and the number of reviews. A compelling case is found in O_3 (*The Nightmare Before Christmas* / *James and the Giant Peach*) which has low reviews but is a costly

article because it is the only one in its context consisting in a two DVD pack. The cluster containing O_2 and O_4 corresponds to Pixar movies, where again O_4 is a three-pack DVD promotion (*Toy story 1-2* and *A Bug's Life*), whereas O_2 corresponds to the poorly reviewed but very expensive *Buzz Lightyear of Star Command: The Adventure Begins* film. The cluster containing O_6 consists of many *Winnie-The-Pooh* related movies with some independent *Tigger* sagas. The anomalous node O_6 (*The Many Adventures of Winnie the Pooh*) is the most expensive film with the highest rate of reviews in this category. The three-nodes cluster containing O_7 (*The Nightmare Before Christmas*) flagged O_7 as an anomaly because its selling price is 3 and 6 times higher than the two other nodes.

Going up in the hierarchy of clusters allows us to detect anomalous nodes in coarser contexts by removing more local outliers. As can be seen in Figure 3.7 (E), the outliers remaining within the light blue cluster, O_3 and O_4 , correspond to the two packs DVD promotions with very high prices, whereas more local anomalies disappear at this scale.

Finally, we see in Figure 3.7 (F) that at the coarsest scale all anomalies disappear except for O_3 (*The Nightmare Before Christmas / James and the Giant Peach*) which remains as a global outlier being not only anomalous in term its attributes but also poorly connected as it has only one connection.

Related methods

As the amount of attributed network data has increased during the last decade, anomaly detection on attributed networks has attracted more attention. Analyzing static graphs for anomaly detection can be grouped in two general classes [5]: anomalies on plain (unlabeled) graphs and anomalies on attributed networks. The former one relates to spotting anomalous nodes regarding abnormal connectivity features in the structure of the network without exploiting any node attribute information [3,4], i.e., isolated or bridge nodes. The second one, into which our method falls, aims to spot anomalous nodes regarding the attributes of nodes within a given local context, i.e., the community of a node or attribute subspace. CODA [66] is a community-based algorithm that uses Hidden Markov Random Fields to characterize both data and links simultaneously turning the outlier detection in an inference problem. Consu+DisOut [156] is a two-step method performing a statistical selection of congruent subspace defined by a subset of node attributes and subsequently rank anomalous nodes

introducing a distance-based outlier model. ALAD [117] is an efficient method to retrieve outlier nodes exploiting network structure and node attribute information jointly considered in a non-negative matrix factorization framework. The RADAR [114] approach proposes a learning framework to identify anomalies via residual analysis, exploring the coherence between attribute information and the network information. AMEN [148] considers ego-networks to characterize anomalous neighbors in attributed networks.

Conclusions and perspectives

In this chapter, we introduce an effective algorithm (MADAN) to perform multi-scale anomaly detection in attributed networks. We characterize anomalous nodes as nodes having a highly concentrated ‘energy’ as a result of smoothing a unit impulse signal centered on each vertex with a heat kernel filter across different graph frequencies. Because anomalous nodes require a notion of context to evaluate the degree of disagreement with respect to the structure and attributes of surrounding nodes, they are intrinsically related to the graph clustering problem.

We exploit the equivalence between filtering a graph signal with a heat kernel and a diffusive process on the network. It turns out that anomalous nodes can be described as nodes with a highly concentrated flow regarding a heat diffusion process. This description allows us to uncover the context of an anomalous node in different scales as the optimal partition uncovered by the Markov stability evaluated at the same time as the filtering operation. This provides a multi-scale description of anomalous nodes, spotting local anomalies in narrow contexts for short diffusion times and gradually more global outliers for increasing times over broader contexts.

We evaluated our approach on synthetic and empirical benchmarks for anomaly detection on attributed graphs. It turns out that MADAN outperforms all considered methods in identifying anomalous when node attributes are defined hierarchically across the network. MADAN also shows competitive performance on real-life networks despite the fact of having arguable ‘ground truth’ labeled datasets.

In addition, the computation of the heat kernel is assessed in synthetic networks, showing an important improvement concerning the exact computation and Padé approximation. Finally, a real-life case of study is presented in order to expose in detail the process of finding robust partitions in the graph

attribute subspace and spotting anomalies in different scales. One of the fundamental features of our method is the definition of the edge weights driving the diffusive flow on the graph. The choice of the similarity function might affect the notion of anomalous nodes and context, especially in high dimensional node attributes, which makes it an exciting topic for further research. A limitation of our approach relies on that we have to manually chose the node features we wish to incorporate in our framework. However, again in high dimensional node's attributes, it is suitable to automatically select the appropriated features for a particular objective, unless the domain expertise needed for a particular application is mastered. This issue remains open for a more in-depth investigation.

Unlike this chapter, the next chapters of the present thesis will cover some machine learning applications in problems defined over a collection of networks.

4

Multi-hop assortativities for networks classification

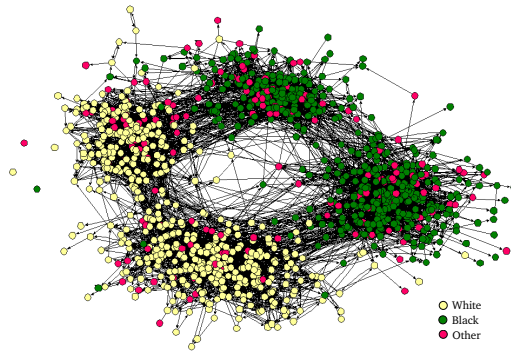


Figure 4.1: A **friendship network at a US school**. Nodes represent high school students and edges a friendship tie. Colors encode the race.

Consider the network of Figure 4.1. A clear pattern emerging from the figure is the division of the network in two groups. It turns out that the division occurs along the line of race: students tend to create social ties with others having the same skin color. The same behavior has been observed and discussed by sociologists in diverse social relationships, such as acquaintances, business, friendship including all kinds of characteristics such as age, income, nationality, language, etc. [130]. The tendency of creating social ties with others perceived as similar in some manner is known as *assortativity mixing* [137] on networks.

In this chapter, we present a generalization of the usual assortativity property (1-hop assortativity) to assortativities between node attributes in diverse scales. We introduce the concept of multi-hop assortativity that captures the similarity of the nodes situated at the extremities of a randomly selected path of a given length. We show that multi-hop assortativity unifies various existing concepts and offers a versatile ‘fingerprint’ to characterize a collection of

networks for performing accurately supervised graph classification tasks.

Introduction

One of the early tasks of network science has been to characterize complex networks through a few global characteristics that summarize their structure in order to understand how different networks, from different fields or with different functional properties behave [138]. For instance, it is well known that many social networks are highly assortative, as people tend to make social connections with others that are similar e.g., in terms of age, race or degree in the network. On the other hand, various biological networks tend to be disassortative, for instance, food web networks [55]. The clustering coefficient, measuring the density of triangles, is also shown to be distinctive characteristics of many social networks.

As the amount of available data on networks keeps growing, one is tempted to use such features systematically in order to correlate them with the context or function attached to the network, using the tools of machine learning, statistics or data mining. One typical task in this context is supervised classification, where we seek to predict for example the toxicity or anti-cancer activity of molecules based on its structure, or which field of physics a researcher contributes to, based on their collaboration ego-network. We speak of supervised classification because an optimal classification criterion is built automatically from a *training* dataset for which the correct answer is known.

In order to apply the powerful supervised classification techniques that have been developed in the last decades (support vector machines, random forests, artificial neural networks, etc.), we might perform a crucial feature engineering process that can be achieved with one of these two strategies. The first one is to transform a given network into a fixed-dimensional vector of numbers characterizing the network (those numbers are called the features of the network); every network now becomes a vector in a Euclidean space (the feature space) so that any classification technique can be used in order to learn which regions of the feature space correspond to toxic molecules, or High Energy Physicists collaboration networks, etc. We call it the feature-based approach. The second strategy defines a quantitative measure of similarity between pairs of networks, called a kernel. This kernel can often be seen as the scalar product between implicit high-dimensional feature representations of a network. The so-called kernel trick allows us to classify networks without ever computing explicitly

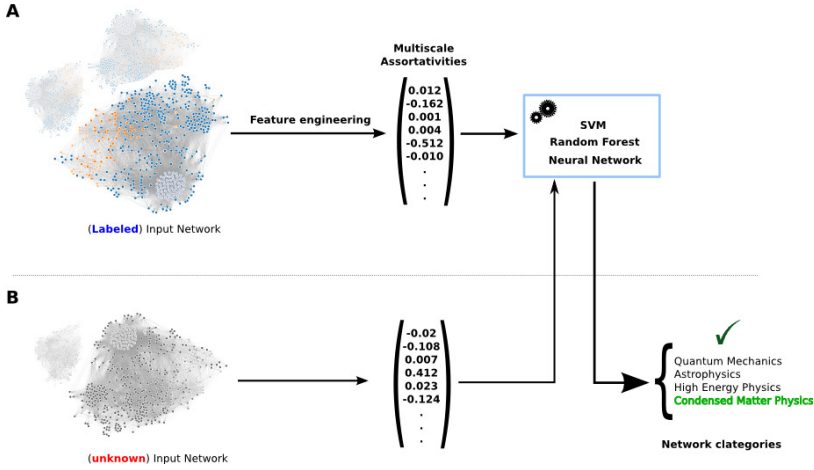


Figure 4.2: **Overview of the proposed approach.** Given a set of networks, we aim to predict the most likely class to which each network belongs. Feature vectors of multi-hop assortativities coefficients are built through a feature engineering process. (A) In the training phase, a subset of labeled networks (for which we know the true class) is used to learn a classifier. They can have nodes metadata (represented by node colors on the picture) or not. (B) The class to which an unseen network belongs is predicted by feeding the trained model with the feature vector representation.

the coordinates of data points in the high-dimensional feature space, sometimes with a substantial gain of computational time over a high-dimensional feature-based approach.

In this Chapter we build on the idea that a dynamical process on a network is strongly indicative of the structure of the underlying graph in which the process takes place, an idea used e.g., in community detection [50] or node centrality [27]. Here, the main idea is to use features associated with the random walk dynamics as network descriptors. Those features are obtained systematically from the correlation patterns of node attributes seen by a random walker at different time instants. The attributes can be either structural (for instance eigenvectors, including the degree) or metadata of the node (e.g., atom type in a molecule network, age or gender in a social network). While such a correlation between two consecutive nodes visited by the random walker coincides with the usual assortativity coefficient, at different times scales one obtains a sequence of *multi-hop assortativities* describing the patterns of average similarity between a node and larger and larger neighborhoods.

The eigenvectors, invaluable source of information when it comes to characterizing a dynamical system, have long been used to explore the structure of the network in terms of centrality (PageRank, degree, eigenvector centrality)

or community detection (spectral methods), and thus form a natural aspect for investigating correlation patterns. Moreover, considering now the ID of the node as its attribute (each node is a category on its own), one will see that a high three-hop assortativity (essentially the probability that a three-hop random walk from a node comes back to the node) is related to the presence of many triangles in the network, hence is akin to a clustering coefficient. The decay to zero of the multi-hop assortativities as the number of hops increases is indicative of the mixing time of the random walker, also indicative of the small-worldness of the network and its heterogeneous structure (existence of communities).

We claim therefore that the multi-hop assortativities generated by a random walker offer a natural, conceptually simple way to generate systematic features acting as a numerical fingerprint for a network, from which an efficient feature-based classification of networks can be performed, see Figure 4.2. It is flexible in that it embeds well-known notions (density of triangles, degree assortativity, etc.) and permits to use categorical (gender, atom type, etc.) or scalar (age, weight, etc.) metadata if available within the same framework.

We evaluate our approach experimentally on social networks and chemoinformatics benchmarks datasets. We compare its classification accuracy concerning some representative graph kernels, neural networks, and features based algorithms of the literature. To make statistical inference from the observed accuracies, we performed a series of robust and straightforward non-parametric tests for comparison of the algorithms [52]. The accuracy measures are, therefore compared using Friedman’s average rank test. Subsequently, Nemenyi’s post hoc test will be employed to test the significance of the differences in rank between individual algorithms.

Our results reveal that generalized assortativities on a small set of elementary structural network features, and any exogenous node metadata (if available), are capable of achieving and outperform in many cases state-of-the-art accuracies, with reasonable computational resources.

This chapter is structured as follows: Section 4.2 introduces multi-hop assortativities. Section 4.3 addresses the problem of definition of features of network dynamics. In section 4.4 we present the experimental setting for assessing our method. We present related approaches in Section 4.5 and finally the discussion in Section 4.6.

Random walks and multi-hop assortativities

In this section, we introduce multi-hop assortativities through a random walk dynamics. For more details about discrete-time random walks, we refer the reader to the section 2.1.5.

Consider an undirected, connected and non-bipartite graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with N vertices and m edges. We will assume for simplicity that the graph is unweighted, but all the results are applicable to nonnegative weights. Extension to directed networks is straightforward as well. The adjacency matrix $\mathbf{A}$ associated to $\mathcal{G}$ is an $N \times N$ binary matrix, with $a_{ij} = 1$ if the vertices i and j are connected, and 0 otherwise. The number of edges in the vertex i is known as the *degree* of the vertex i , denoted d_i . Then the degree vector of $\mathcal{G}$ can be compiled as $\mathbf{d} = \mathbf{A}\mathbf{1}$, where $\mathbf{1}$ is a $N \times 1$ ones vector. For later computations, we define also the $N \times N$ degree matrix $\mathbf{D} = \text{diag}(\mathbf{d})$.

The standard random walk on such a graph defines a Markov chain in which transition probabilities are split uniformly among the edges, with a transition probability $1/d_i$:

$$\mathbf{p}_{t+1} = \mathbf{p}_t[\mathbf{D}^{-1}\mathbf{A}] \equiv \mathbf{p}_t\mathbf{M} \quad (4.1)$$

Here $\mathbf{p}_t$ is the $1 \times N$ normalized probability distribution of the random walker over the nodes at time t , and $\mathbf{M}$ the $N \times N$ transition matrix. Under the assumptions on the graph (connected, undirected, and non-bipartite), the dynamics converges to a unique *stationary distribution* vector $\boldsymbol{\pi} = \mathbf{d}^T/2m$. We define also the $N \times N$ diagonal matrix $\boldsymbol{\Pi} = \text{diag}(\boldsymbol{\pi})$.

Consider a scalar node attribute such as the degree, centrality, age (in a social network), etc. It is known that many real networks exhibit interesting assortativity properties, i.e., a high covariance between the two end node's attributes of a randomly selected edge. For the random walker in stationary distribution, this translates into the covariance of the node attribute visited at two consecutive time instants τ and $\tau + 1$, since the walker visits each edge of the graph with the same frequency.

This suggests to consider *multi-hop assortativities*, as the covariance between nodes attributes seen by the walker at times τ and $\tau + t$, for any $t = 0, 1, 2, 3, \dots$. While the case $t = 0$ is simply the variance, the case $t > 1$ explores assortativity patterns within multi-hop neighborhoods, i.e., the covariances between extremities of a randomly selected path of length t .

More formally, we consider the N -dimensional vector $\mathbf{X}_t$ as a random

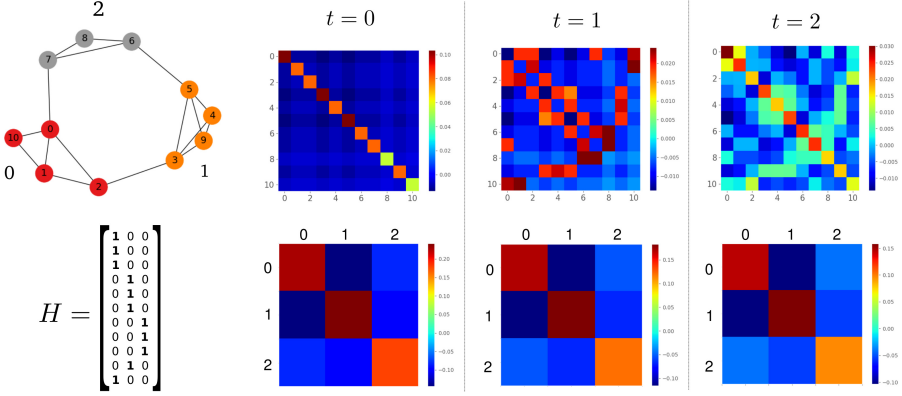


Figure 4.3: **Autocovariance of a categorical attribute.** Top matrices correspond to the autocovariance matrices (Eq. 4.2) of a random walk dynamics on the left hand graph. Categorical node attributes are represented by colors on the graph. They define a partition encoded in the binary matrix H . Bottom matrices correspond to the category-level autocovariance matrix $H^T \rho(t) H$ for different time instants. In particular, the diagonal entries are $r(t, \mathbf{h}_i)$, see Eq. 4.4, and the trace of the matrix is $r(t, H)$, see Eq. 4.6.

indicator vector of the presence of a random walker in time such that for the k th entry $\mathbf{X}_t(k) = 1$ if the walker is at node k at time t , and zero otherwise. Thus, the *autocovariance* matrix [50] of the observable process at time t is expressed as

$$\rho(t) = \text{Cov}(\mathbf{X}_\tau, \mathbf{X}_{\tau+t}) = \Pi M^t - \Pi^T \Pi \quad (4.2)$$

Compiling the scalar nodes attributes in a $N \times 1$ attribute vector $\mathbf{v}$, the covariance of the attribute $\mathbf{v}$ is given by

$$r(t, \mathbf{v}) = \mathbf{v}^T \rho(t) \mathbf{v} \quad (4.3)$$

as indeed the value of the node attribute observed by the random walker on the node it stands on at time τ is none but the random variable $\sum v_k \mathbf{X}_\tau(k)$ summed over all nodes k .

We call $r(t, \mathbf{v})$ the t -hop *assortativity* of a numerical attribute $\mathbf{v} \in \mathbb{R}^{N \times 1}$. On an undirected network, evaluating $t = 1$ will coincide with the usual assortativity of a scalar attribute, except that the latter often normalized as a correlation. We choose not to normalize it as it increases the dynamical range of the assortativity, hence its discriminating power to distinguish a network from another. For $t = 0$ it is simply the variance of attribute $\mathbf{v}$ among the nodes, with each node's attribute weighted proportionally to the degree of the

node.

Note that as the number t of hops grows to infinity, $\boldsymbol{\rho}(t)$ converges to the zero matrix for a connected non-bipartite network, as every row of $\mathbf{M}^t$ converges to $\boldsymbol{\pi}$. Therefore all multi-hop assortativities tend to zero with the number of hops, and the rate of convergence is well-known to be given by the spectral gap of $\mathbf{M}$. The spectral gap is the inverse of the mixing time of the random walk, indicative of an effective typical diameter of the graph, and equivalent through Cheeger's inequality to the existence of a strong community structure in the network [9, 39]. Note that if the network is connected but bipartite, a unique stationary vector $\boldsymbol{\pi}$ is still defined and $\boldsymbol{\rho}(t)$ can still be constructed but oscillates periodically as the number t of hops grows.

It often happens that a node attribute is categorical rather than numerical, e.g., gender or political parties in a social network or atom type in chemical compounds. In this case, we compute an assortativity coefficient by encoding each category as a binary characteristic vector. One therefore encode a k -category attribute as an $N \times k$ binary matrix $\mathbf{H}$, every row which contains exactly one 1 (one-hot encoding). We compute the autocovariance of each category, i.e. each column $\mathbf{h}_i$ of $\mathbf{H}$ as

$$r(t, \mathbf{h}_i) = \mathbf{h}_i^T \boldsymbol{\rho}(t) \mathbf{h}_i \quad (4.4)$$

with $1 \leq i \leq k$, yielding a k -dimensional vector for each time t .

It can also be written directly in terms of probability as follows [50, 106]

$$r(t, \mathbf{h}_i) = \text{Proba}(x_\tau \text{ and } x_{\tau+t} \text{ both in category } i) - \text{Proba}(x_\tau \text{ and } x_{\tau+\infty} \text{ in category } i) \quad (4.5)$$

where x_τ is the node visited by the random walker at time τ . In other words, it measures how likely it is that a randomly selected node and a randomly selected t -hop neighbour both belong to category i . We remove the term corresponding to infinite delay $t \rightarrow \infty$, which is also the probability $(\boldsymbol{\pi} \mathbf{h}_i)^2$ that two independent random walkers belong to category i .

In the case of many categories, it is convenient to characterize the autocovariance of a given node labeling with a given time delay with a single number, and therefore we sum the autocovariances of each category:

$$r(t, \mathbf{H}) = \sum_{i=1}^k r(t, \mathbf{h}_i) = \text{Tr}[\mathbf{H}^T \boldsymbol{\rho}(t) \mathbf{H}] \quad (4.6)$$

It can also be written directly in terms of probability as

$$r(t, \mathbf{H}) = \frac{\text{Proba}(x_t \text{ and } x_{\tau+t} \text{ in same category}) - \text{Proba}(x_t \text{ and } x_{\tau+\infty} \text{ in same category})}{\sum_i (\pi \mathbf{h}_i)^2} \quad (4.7)$$

In other words, it measures how frequently a randomly selected node and a randomly selected t -hop neighbor belong to the same category. We remove the term corresponding to infinite delay $t \rightarrow \infty$, which is also the probability $\|\pi \mathbf{H}\|_2^2$ that two independent random walkers belong to the same category, which is $\sum_i (\pi \mathbf{h}_i)^2$.

For $t = 1$ this is the usual, unnormalized assortativity coefficient of the categorical attribute [42]. For $t = 0$, this coefficient does not depend on the structure of the network, only of the relative frequencies of the categories, taking a high value $1 - \frac{1}{k}$ if the k categories are equally spread, in terms of total degrees of nodes of each category. Uneven categories lead to a variance close to zero. This measure of diversity is essentially the Rényi entropy of the node attribute [72], or Simpson's diversity index used in ecology or economics [169]. For general t , this coefficient is the *multi-hop assortativity* of the categorical node attribute encoded by the matrix $\mathbf{H}$, and captures the tendency of nearby nodes to have the same categorical attribute.

It can be noted that a categorical attribute $\mathbf{H}$ defines a partition of the nodes, and $r(1, \mathbf{H})$ happens to coincide with the modularity of the partition [139] used in community detection, or graph clustering. The quantity $r(t, \mathbf{H})$, called in this context Markov stability [50], allows multi-scale community detection by scanning through the time parameter t .

Features of network dynamics

We have seen how to define multi-hop assortativity coefficients through a random walk dynamics on networks. We argue that multi-hop assortativity patterns of structural and metadata (if available) node attributes emerge as a useful fingerprint of the network, characterizing the interaction of nodes attributes in many levels and discriminating between diverse network structures. We aim to build a strong feature vector representation of networks based on covariances of node attributes.

Given that we have on top of a network a stationary random walk dynamics, it is natural to consider its spectrum. Indeed, the eigenvectors of matrices describing random walks on graphs have been used extensively for many graph

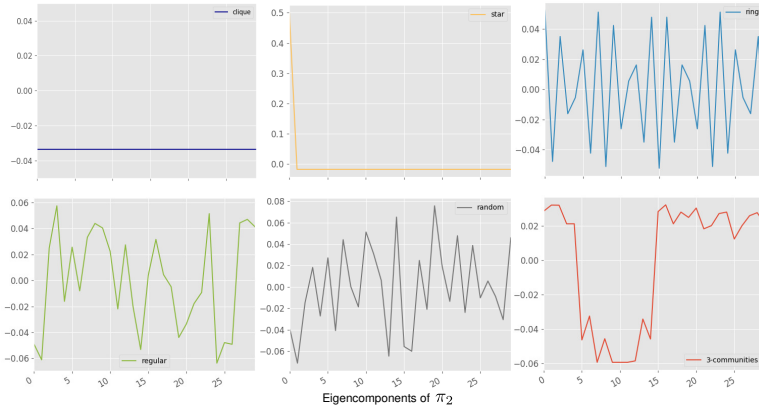


Figure 4.4: **Eigenvectors as network features.** We plot the second dominant eigenvector (π_2) of $\mathbf{M}$ of six different network topologies with 30 nodes each: (1) clique, (2) star, (3) ring, (4) random 8-regular graph (a graph picked randomly uniformly among all undirected graphs with all nodes of degree 8), (5) Erdős-Rényi random graph with $p = 0.4$ (6) network with a planted partition of 3 communities with 5, 10 and 15 nodes, within-group probability of 0.8 and between-group probability of 0.02.

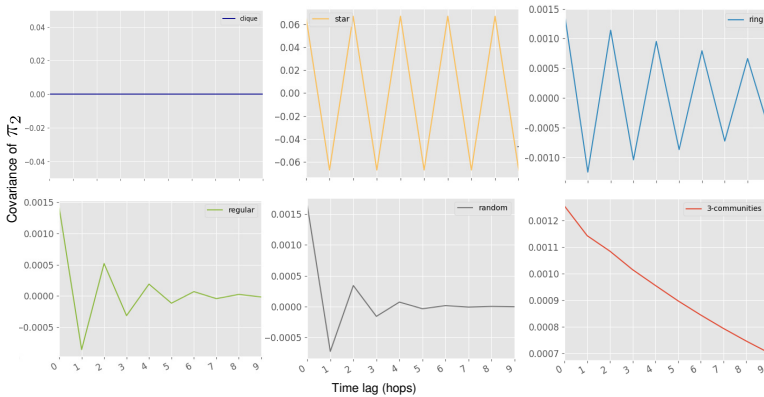


Figure 4.5: **Autocovariances as network features.** We plot $r(t, \mathbf{v})$ (Eq. 4.3) with $\mathbf{v}$ as the second dominant eigenvector (π_2) of the transition matrix $\mathbf{M}$ vs time, on six different network topologies with 30 nodes each: (1) clique, (2) star, (3) ring, (4) random 8-regular graph (a graph picked randomly uniformly among all undirected graphs with all nodes of degree 8), (5) Erdős-Rényi random graph with $p = 0.4$ (6) network with a planted partition of 3 communities with 5, 10 and 15 nodes, within-group probability of 0.8 and between-group probability of 0.02.

mining purposes, i.e., spectral clustering [96, 141], eigenvector centrality [20], Pagerank centrality [27] and so on. Moreover, the Pagerank, the first left eigenvector of the random walk matrix $\mathbf{M}$, coincides in this undirected case with the degree (normalized by the sum of all degrees), and the degree assor-

tativity has been one of the earliest and most important structural properties considered in networks science [138]. It is therefore natural to check if multi-hop assortativities of several leading eigenvectors constitute an appropriated set of features characterizing the structure of the graph.

Let us illustrate this fact in one example. As can be seen on Figure 4.4, choosing a network attribute such as the second left eigenvector of $\mathbf{M}$, yields a reasonable signature to discriminate among different network structures. However, its assortativity, see Figure 4.5, has additional advantages: it is an adequate fingerprinting of the network in terms of structural differentiation among topologies but also is a lower-dimensional representation of the same network. In Figure 4.5 each dimension correspond to an assortativity coefficient of the selected node attribute (second dominant eigenvector of M) in a given scale Eq. 4.3.

The simplest node attribute we can think about is certainly the node ID itself, i.e. $\mathbf{H} = \mathbf{I}$ (identity matrix) in Eq. 4.6. It is straightforward that for $t = 0$ it yields an assortativity $r(0, \mathbf{I}) = 1 - \|\boldsymbol{\pi}\|^2$, delivering effectively a variance of the degree distribution. For $t = 3$, it counts essentially, the probability for the random walker to follow a triangle, thus representing a variant of the global clustering coefficient, yet another structural feature of interest. For arbitrary $t > 0$ it counts essentially the density of t -cycles.

In addition to second-moment measures given by multi-hop assortativities, it is of course natural to include first moments, i.e. averages, in the list of features able to characterize a network. Any scalar attribute $\mathbf{v} \in \mathbb{R}^{N \times 1}$ generates the network feature $\boldsymbol{\pi}^T \mathbf{v} = \sum_k \pi_k v_k$ (average weighted by the degrees), while a categorical attribute encoded by the membership matrix $\mathbf{H}$ generates the list of respective frequencies of each category, $\boldsymbol{\pi}^T \mathbf{H} \in \mathbb{R}^{1 \times k}$.

Certainly real life networks are more complex and heterogeneous than our toy example. In complex networks, it is natural to find diversity in mixing patterns arise from diverse nodes attributes. For our experiments, we opt for selecting a reduced set of features capturing local and global structural mixing properties as well as metadata node attributes:

1. Multi-hop assortativities of node IDs, setting $\mathbf{H} = \mathbf{I}$ in Eq. 4.6 for $0, 1, \dots, t$ hops
2. Average of first p dominant left eigenvector of $\mathbf{M}$,
i.e. $\{\boldsymbol{\pi}^T \mathbf{v} : \mathbf{v} \text{ is a dominant eigenvector of } \mathbf{M}\}$
3. Multi-hop assortativities of the first p dominant left eigenvectors of $\mathbf{M}$

(Eq. 4.3) for $0, 1, \dots, t$ hops

4. (If available) average of categorical metadata node attributes: $\{\pi^T \mathbf{h}_i : \mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_k], 1 \leq i \leq k\}$
5. (If available) multi-hop assortativities of categorical metadata node attributes (Eq. 4.4) for $0, 1, \dots, t$ hops
6. Number of nodes
7. Number of edges

We compile the aforementioned features in a single feature vector that will be used as a network fingerprint for data mining purposes. Its dimension varies according with the choice of eigenvectors $\mathbf{p}$, number of different node categories k the dataset has and the number t of hops for multi-hop assortativities. Our experiments were performed with $t = 0, 1, 2, 3$, keeping the first three dominant left eigenvectors of $\mathbf{M}$. For clarity, we emphasize that features number 1, do not require the networks under consideration to share the same size or a particular node ordering, as assortativity of a given number of hops is a single number summing the individual node assortativities, independently of node names or ordering.

Note that taking the left eigenvectors of $\mathbf{M}$ as node features is more convenient because its simplicity: the stationary distribution of the associated dynamics is proportional to the node degree. Moreover, we stress that the *right eigenvectors* of $\mathbf{M}$ [63] embed well known graph metrics which could potentially improve the performance of our approach. For instance, the solution of the normalized cuts [165] can be stated as the smallest non-trivial right eigenvector of $\mathbf{I} - \mathbf{M}$. Markov diffusion maps [43] are related with the right eigenvectors of $\mathbf{M}$ associated with its decreasingly sorted eigenvalues. Correspondence analysis [73] considers the largest nontrivial right eigenvalue of $\mathbf{M}^2$ sorted in decreasing order of eigenvalue modulus. In the undirected case, left and right eigenvectors of $\mathbf{M}$ are related up to degree normalization. This issue is furthermore stated as a clear direction for future research.

In the next section, we evaluate our method empirically on many real-life network datasets and compare them against other approaches of the literature.

Table 4.1: Statistics for benchmark datasets. First six correspond to chemoinformatic and the rest to social network datasets. UN means unlabeled nodes

Dataset	# graphs	classes	node attribs	avg nodes	avg edges
MUTAG	188	2	7	17.93	19.79
PTC	344	2	19	14.29	14.69
NCI1	4110	2	22	29.87	32.3
NCI109	4127	2	19	29.68	32.13
ENZYMES	600	6	3	32.63	62.14
PROTEINS	1113	2	3	39.06	72.82
COLLAB	5000	3	UN	74.49	2457.78
REDDIT-BINARY	2000	2	UN	429.63	497.75
REDDIT-MULTI-5K	5000	5	UN	508.52	594.87
REDDIT-MULTI-12K	11929	11	UN	391.41	456.89
IMDB-BINARY	1000	2	UN	19.77	96.53
IMDB-MULTI	1500	3	UN	13.0	65.94

Applications in Graph Classification: experiments and results

Our experiments aim to show that multi-hop assortativities are useful descriptors for real-life networks discriminating well between classes. The experiments are conducted on networks with and without node metadata. The classification consists of predicting the most likely label to an unseen input example. To do so, we train two popular classifiers widely used in the literature: Support Vector Machines (SVM) [23] and Random Forest [25] (RF). The former consists of learning an optimal hyperplane maximizing the margin of separation between the data points in the feature space. On the other hand, a random forest classifier creates a set of decision trees predictors from a randomly selected subset of the training set. It aggregates the votes from different decision trees to decide the final class of a test example.

Details about the datasets and experimental setup are explained in the following subsections.

Datasets

Twelve real-world datasets were used in our experiments. We summarize their general properties in Table 4.1. The first six correspond to the chemoinfor-

matic category, composed by undirected graphs representing either molecules, enzymes or proteins. They have all categorical node attributes encoding a particular functional property related its function. The later six are unlabeled node (i.e without node attribute) social network datasets. We will describe them below.

MUTAG [48] is a nitro compounds dataset divided in two classes according to their mutagenic activity on bacterium *Salmonella Typhimurium*. **PTC** dataset [90] contains compounds labeled according to carcinogenicity on rodents, divided in two groups. Vertices are labeled by 19 atom types. The **NC11** and **NCI109** [185] from the National Cancer Institute (NCI), are two balanced dataset of chemical compounds screened for activity against non-small cell lung cancer and ovarian cancer cell. They have 22 and 19 categorical node labels respectively. **PROTEINS** [22], [160] is a two-class dataset in which nodes are secondary structure elements (SSEs). Nodes are connected if they are contiguous in the aminoacid sequence. **ENZYMES** [22], [160] is a dataset of protein tertiary structures consisting of 600 enzymes from the BRENDA enzyme database. The task is to assign each enzyme to one of the 6 EC top-level classes.

From the social networks pool [195], **COLLAB** is a scientific collaboration dataset, where ego-networks of researchers that have worked together are constructed. The task is to determine whether the ego-collaboration network belongs to any of three classes, namely, *High Energy Physics*, *Condensed Matter Physics* and *Astro Physics*. **REDDIT-BINARY**, **REDDIT-MULTI-5K** and **REDDIT-MULTI-12K** are three balanced datasets having two, five and eleven groups respectively. Each one contains a set of graphs representing an on-line discussion thread where nodes correspond to users, and there is an edge between them if anyone responds to another’s comment. The task is then to discriminate between threads from which the subreddit was originated. **IMDB-BINARY** is a dataset of ego-networks of actors that have appeared together in any movie. Graphs are constructed from *Action* and *Romance* genres. The task is to identify to which genre an ego-network graph belongs. **IMDB-MULTI** is the same, but consider three movie genres: *Comedy*, *Romance* and *Sci-Fi*.

Experimental setup

In order to be as fair as possible, we follow the experimental setup of [195] and [163] and [13]. We assess the performance of our method against some

representative graph kernels, feature-based, and neural networks techniques of the literature. The algorithms to which we compare are the Graphlet [163], Shortest path [21] and the Weisfeiler-Lehman subtree kernels [163], as well as their respective deep versions [195]. Random walks based kernels as p -step random walk [171], the random walk [67] and Ramon & Gartner kernels [152] are also considered. We also compare against the feature-based method [13], the convolutional neural network PSCN [142] and the shift aggregate extract network (SAEN) [144].

For the graph-kernel methods, we used the Matlab scripts¹. The Deep Graph kernel scripts (in Python) were taken from one of the author’s website². We coded the feature-based approach [13] and our algorithm in Matlab. We made available our source code in this site³.

Each dataset is split randomly in training and testing sets. The best C-SVM and Random Forest model is selected over a 10-fold cross-validation scheme. The parameter C and the number of trees are optimized only on the training set. Thus, we report the generalization accuracy on the unseen test set. In order to exclude the random effect of the data splitting, we repeated the whole experiment 10 times. Finally, we report the *average prediction accuracies* and its *standard deviation*.

The parameters for the graph-kernels approaches are also cross-validated on the training set following [163] and [195] settings. The details about the parameters can be found in the Appx. C.1.

For each kernel, we report the result for the parameter that achieves the best classification accuracy. For the feature-based approach [13], feature vectors were built with the same network features they reported in their paper: number of nodes, number of edges, average degree, degree assortativity, number of triangles and global clustering coefficient. Finally, for PSCN and SAEN, we compare with the accuracies reported in [142] and [144], respectively.

Regarding our method, Table 4.2 shows the features from Section 4.3 we used in our experiments. The last column shows the length of the feature vector for the graphs within each dataset.

¹<http://mlcb.is.tuebingen.mpg.de/Mitarbeiter/Nino/Graphkernels/>

²http://www.mit.edu/~pinary/kdd/DEEP_GRAPH_KERNELS_CODE.tar.gz

³<https://github.com/leoguti85/MaF>

Table 4.2: Input features for each dataset

Dataset	Features	Num of eigenvectors	Node labels	Dim
MUTAG	[1,2,3,4,5,6,7]	3	7	56
PTC	[1,2,3,4,5,6,7]	3	19	116
NCI1	[1,2,3,4,5,6,7]	3	22	131
NCI109	[1,2,3,4,5,6,7]	3	19	116
ENZYMES	[1,2,3,4,5,6,7]	3	3	36
PROTEINS	[1,2,3,4,5,6,7]	3	3	36
COLLAB	[1,2,3,6,7]	5	-	31
REDDIT-BINARY	[1,2,3,6,7]	5	-	31
REDDIT-MULTI-5K	[1,2,3,6,7]	5	-	31
REDDIT-MULTI-12K	[1,2,3,6,7]	5	-	31
IMDB-BINARY	[1,2,3,6,7]	3	-	21
IMDB-MULTI	[1,2,3,6,7]	3	-	21

Results

We test our method on all considered benchmark datasets and compare our classification accuracies with the ones achieved by the aforementioned algorithms. We report three instances of our method: **MaF-SVM** when we train using Support Vector Machines and **MaF-RF** when we use Random Forest classifier on our multi-hop assortativities features (MaF). **MaF-nolab** corresponds to a Random Forest on a subset of features discarding node metadata information explicitly.

For the experiments on **social network** datasets we apply those algorithms that can handle unlabeled node graphs. Results for social graphs are depicted below in Table 4.3 (*graph-kernel methods*) and Table 4.4 (*neural nets and feature-based approach*).

As can be seen from Table 4.3 we outperform all graph kernel methods on all social networks except WL on IMDB-MULTI, while being comparable with IMDB-BINARY. In particular, our method performs better on datasets with large networks and number of examples, see REDDIT and COLLAB in Table 4.3. Regarding Table 4.4, our method generally performs better than Convolutional Neural Networks (PSCN), while remaining comparable with Feature-Based (FB), which is expected because both are methods of the same nature. However, SAEN outperforms our method in IMDB and REDDIT multi-class problems. Experiments suggest that in general, MaF-SVM is more accurate

Table 4.3: *Graph-kernel methods on social networks*: Mean and standard deviation of classification accuracy for Random Walk (RW) [67], Weisfeiler-Lehman (WL) [163], Graphlet (GK) [164], Deep Graphlet kernels (DGK) [195], multi-hop assortativities (MaF) (our method)

Dataset	RW	WL	GK	DGK	MaF-SVM	MaF-RF
COLLAB	69.01 $\pm$ 0.09	77.79 $\pm$ 0.19	72.84 $\pm$ 0.28	73.09 $\pm$ 0.25	75.54 $\pm$ 1.38	78.24 $\pm$ 1.57
IMDB-BIN	64.54 $\pm$ 1.22	72.86 $\pm$ 0.76	65.87 $\pm$ 0.98	66.96 $\pm$ 0.56	71.30 $\pm$ 3.23	71.60 $\pm$ 4.45
IMDB-MULT	34.54 $\pm$ 0.76	50.55 $\pm$ 0.55	43.89 $\pm$ 0.38	44.55 $\pm$ 0.52	47.53 $\pm$ 3.24	45.20 $\pm$ 3.54
REDDIT-BIN	67.63 $\pm$ 1.01	69.57 $\pm$ 0.88	77.34 $\pm$ 0.18	78.04 $\pm$ 0.39	89.00 $\pm$ 2.25	88.90 $\pm$ 2.20
REDDIT-M5K	> 72h	47.72 $\pm$ 0.48	41.01 $\pm$ 0.17	41.27 $\pm$ 0.18	54.37 $\pm$ 2.08	51.39 $\pm$ 1.91
REDDIT-M12K	> 72h	38.47 $\pm$ 0.12	31.82 $\pm$ 0.08	32.22 $\pm$ 0.10	44.52 $\pm$ 1.44	43.50 $\pm$ 1.03

Table 4.4: *Neural nets and feature-based approaches on social networks*: Mean and standard deviation of classification accuracy for Shift aggregate extract network (SAEN) [144], Convolutional Neural Network (PSCN) [142], Feature-Based (FB) [13], multi-hop assortativities (MaF) (our method)

Dataset	SAEN	PSCN	FB	MaF-SVM	MaF-RF
COLLAB	75.63 $\pm$ 0.31	72.60 $\pm$ 2.15	76.35 $\pm$ 1.64	75.54 $\pm$ 1.38	78.24 $\pm$ 1.57
IMDB-BINARY	71.26 $\pm$ 0.74	71.00 $\pm$ 2.29	72.02 $\pm$ 4.71	71.30 $\pm$ 3.23	71.60 $\pm$ 4.45
IMDB-MULTI	49.11 $\pm$ 0.64	45.23 $\pm$ 2.84	47.34 $\pm$ 3.56	47.53 $\pm$ 3.24	45.20 $\pm$ 3.54
REDDIT-BINARY	86.08 $\pm$ 0.53	86.30 $\pm$ 1.58	88.98 $\pm$ 2.26	89.00 $\pm$ 2.25	88.90 $\pm$ 2.20
REDDIT-MULTI-5K	52.24 $\pm$ 0.38	49.10 $\pm$ 0.70	50.83 $\pm$ 1.83	54.37 $\pm$ 2.08	51.39 $\pm$ 1.91
REDDIT-MULTI-12K	46.72 $\pm$ 0.23	41.32 $\pm$ 0.42	42.37 $\pm$ 1.27	44.52 $\pm$ 1.44	43.50 $\pm$ 1.03

Table 4.5: *Methods that do not exploit node labels on chemoinformatic graphs*: Mean and standard deviation classification accuracy for: Graphlet (GK) [164], Deep Graphlet (DGK) [195], Feature-Based (FB) [13]) and our multi-hop assortativities Features, without using node labels (MaF-nolab) and with labels (MaF-SVM, MaF-RF)

Data	GK	DGK	FB	MaF-nolab	MaF-SVM	MaF-RF
MUTAG	81.66 $\pm$ 2.11	82.66 $\pm$ 1.45	84.66 $\pm$ 2.01	82.48 $\pm$ 9.28	85.09 $\pm$ 7.34	89.89 $\pm$ 5.58
PTC	57.26 $\pm$ 1.41	57.32 $\pm$ 1.13	55.58 $\pm$ 2.30	61.99 $\pm$ 7.06	58.79 $\pm$ 7.11	61.34 $\pm$ 7.61
NCI1	62.28 $\pm$ 0.29	62.48 $\pm$ 0.25	62.90 $\pm$ 0.96	70.12 $\pm$ 1.58	73.89 $\pm$ 1.48	77.32 $\pm$ 1.68
NCI109	62.60 $\pm$ 0.19	62.69 $\pm$ 0.23	62.43 $\pm$ 1.13	67.87 $\pm$ 2.15	73.49 $\pm$ 1.82	74.97 $\pm$ 2.19
PROTEINS	71.67 $\pm$ 0.55	71.68 $\pm$ 0.50	69.97 $\pm$ 1.34	73.05 $\pm$ 3.35	75.20 $\pm$ 2.67	76.73 $\pm$ 2.97
ENZYMES	26.61 $\pm$ 0.99	27.08 $\pm$ 0.79	29.00 $\pm$ 1.16	40.67 $\pm$ 3.96	48.17 $\pm$ 6.60	56.17 $\pm$ 9.10

for the multi-class problems than MaF-RF.

On the other hand, results of **chemoinformatic datasets** are depicted in Table 4.5 (*Methods that do not exploit node metadata*), Table 4.6 (*random-walks and Ramon & Gartner kernel*), and Table 4.7 (*others graph-kernels and neural nets approaches*)

In Table 4.5 we compare against methods cannot handle node metadata. As we can see, our method (MaF-nolab) based solely on the structure of the graphs

Table 4.6: *Random-walks and Ramon & Gartner kernels on chemoinformatic graphs*: Mean and standard deviation classification accuracy for: Ramon & Gartner (RG) [152], p-Random Walk (pRW) [171], Random Walk (RW) [67]) and our multi-hop assortativities, without using node labels (MaF-nolab) and with labels (MaF-SVM, MaF-RF)

Data	RG	pRW	RW	MaF-nolab	MaF-SVM	MaF-RF
MUTAG	84.88 $\pm$ 1.86	80.05 $\pm$ 1.64	83.72 $\pm$ 1.50	82.48 $\pm$ 9.28	85.09 $\pm$ 7.34	89.89 $\pm$ 5.58
PTC	58.47 $\pm$ 0.90	59.38 $\pm$ 1.66	57.85 $\pm$ 1.30	61.99 $\pm$ 7.06	58.79 $\pm$ 7.11	61.34 $\pm$ 7.61
NCI1	56.61 $\pm$ 0.53	> 72h	48.15 $\pm$ 0.50	70.12 $\pm$ 1.58	73.89 $\pm$ 1.48	77.32 $\pm$ 1.68
NCI109	54.62 $\pm$ 0.23	> 72h	49.75 $\pm$ 0.60	67.87 $\pm$ 2.15	73.49 $\pm$ 1.82	74.97 $\pm$ 2.19
PROTEINS	70.73 $\pm$ 0.35	71.16 $\pm$ 0.35	74.22 $\pm$ 0.42	73.05 $\pm$ 3.35	75.20 $\pm$ 2.67	76.73 $\pm$ 2.97
ENZYMES	16.96 $\pm$ 1.46	30.01 $\pm$ 1.01	24.16 $\pm$ 1.64	40.67 $\pm$ 3.96	48.17 $\pm$ 6.60	56.17 $\pm$ 9.10

Table 4.7: *Other graph-kernels and neural nets approaches on chemoinformatic graphs*: Deep Shortest-path (DSP) [195], Weisfeiler-Lehman (WL) [163], Deep Weisfeiler-Lehman (DWL) [195] kernels, Convolutional Neural Network (PSCN) [142]) and multi-hop assortativities (MaF-SVM, MaF-RF) (our method)

Data	DSP	WL	DWL	PSCN	MaF-SVM	MaF-RF
MUTAG	87.44 $\pm$ 2.72	80.72 $\pm$ 3.00	82.94 $\pm$ 2.68	92.63 $\pm$ 4.21	85.09 $\pm$ 7.34	89.89 $\pm$ 5.58
PTC	59.52 $\pm$ 2.19	56.97 $\pm$ 2.01	59.17 $\pm$ 1.56	60.00 $\pm$ 4.82	58.79 $\pm$ 7.11	61.34 $\pm$ 7.61
NCI1	73.55 $\pm$ 0.51	80.13 $\pm$ 0.50	80.31 $\pm$ 0.46	78.59 $\pm$ 1.89	73.89 $\pm$ 1.48	77.32 $\pm$ 1.68
NCI109	73.26 $\pm$ 0.26	80.22 $\pm$ 0.34	80.32 $\pm$ 0.33	–	73.49 $\pm$ 1.82	74.97 $\pm$ 2.19
PROTEINS	75.78 $\pm$ 0.54	72.92 $\pm$ 0.56	73.30 $\pm$ 0.82	75.89 $\pm$ 2.76	75.20 $\pm$ 2.67	76.73 $\pm$ 2.97
ENZYMES	41.65 $\pm$ 1.57	53.15 $\pm$ 1.14	53.43 $\pm$ 0.91	–	48.17 $\pm$ 6.60	56.17 $\pm$ 9.10

outperforms in all datasets GK, DGK and FB algorithms. Similarly, comparing against graph kernel methods in Table 4.6, our approach exhibits the best performance along all considered datasets. Even in the case where we do not take node metadata into account our approach has remarkable performances. Indeed, including node metadata into our algorithm improves in general our classification accuracy. We see also in Table 4.7 that in the multi-class dataset (ENZYMES) our approach performs the best being consistent with the results on social networks. Looking in particular at the proteins graphs (PTC, PROTEINS), our method outperforms other approaches. Meanwhile, unlike for social network datasets our method is outperformed by the deep WL and WL kernels on NCI graphs see Table 4.7.

In order to state the significance of our method, we use a statistical analysis (appendix B) to compare the accuracies of the different algorithms. We applied it on chemoinformatic and social networks datasets independently, including only algorithms with complete scores along datasets, e.g. full-filled columns on Tables 4.3, 4.4, 4.5, 4.6 and 4.7.

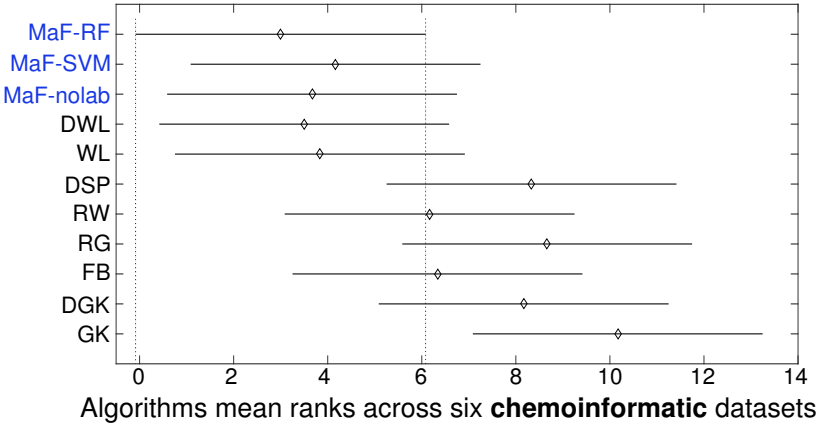


Figure 4.6: Mean rank (diamonds) and critical difference (dashed line) of the Friedman/Nemenyi test on chemoinformatic datasets. The critical difference is 6.1633. Our method is in blue.

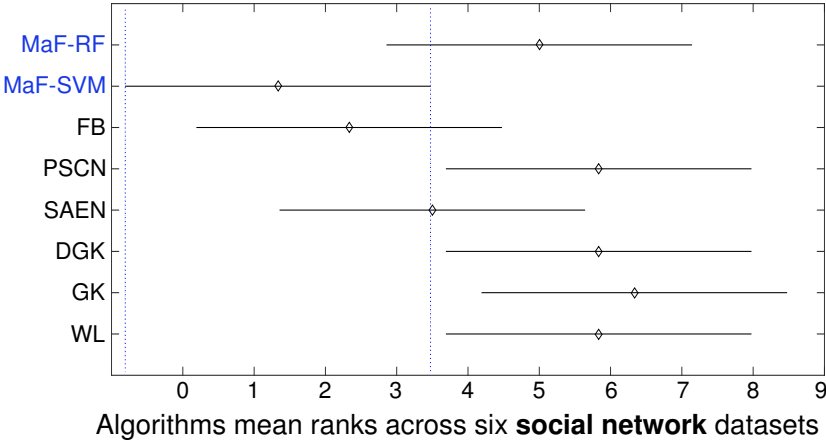


Figure 4.7: Mean rank (diamonds) and critical difference (dashed line) of the Friedman/Nemenyi test on social network datasets. The critical difference is 4.2863. Our method is in blue.

For each group, the Friedman chi-square statistic (Eq. B.1) and corresponding p -values were computed. Indeed, $Q = 34.63$ with corresponding p -value of 0.000144 are reported for chemoinformatic datasets and $Q = 24.66$ with p -value of 0.000869 for the social network benchmark.

As this test informs that some algorithms have significantly different performances than others ($p < 0.005$), a post hoc Nemenyi test was applied to each class distribution. The following significance diagrams (Figures 4.6 and

4.7) display the accuracy performance ranks of the algorithms along with the Nemenyi’s critical difference (CD) tail (Eq. B.2). The CD value for social network benchmarks was 4.2863 and 6.1633 for chemoinformatics. Each diagram shows the algorithms used on each benchmark, listed in ascending order of ranked performance on the y -axis, and the algorithm’s mean rank across all six datasets (Table 4.1), displayed on x -axis. The two vertical dashed lines indicate the start and the end of the best performing method.

Regarding chemoinformatic benchmarks in the Figure 4.6, it shows that our method MaF-RF is the best performing technique with an average ranking of 3.0. The diagram clearly shows that the Graphlet kernel (GK) perform significantly lower than the best performing algorithm, with an AR of 10.166. We can confirm our previous observation in which exploiting node attributes improves the performance of the task.

Concerning the social network benchmarks, Figure 4.7 shows that our MaF-SVM method performs the best with an AR of 1.333. This diagram clearly shows that Weisfeiler-Lehman (WL), Graphlet (GK), Deep Graphlet kernels as well as the convolutional neural network (PSCN) perform significantly worse than the best performing technique, with AR values of 5.833, 6.333, 5.833 and 5.833 respectively.

Computational Cost

Computing our assortativity features relies principally on two aspects: eigenvector computations and matrix-vector multiplication (e.g., covariance matrix times eigenvectors). In order to make fair comparisons, for all methods, we report the runtime for features generation. Therefore, we report the runtime for graph-kernel matrix computation, feature vector generation for FB and the multi-hop assortativity features formed with the attributes enumerated in Section 4.3. The results are shown in Table 4.8.

Our method (MaF) is clearly faster than the shortest path (SP) and deep shortest path (DSP) kernels which do not work on unlabeled node networks (NA). We also outperform Graphlet (GK) and Deep Graphlet (DGK) kernels on chemoinformatic and small size datasets. Among the graph-kernel methods, Weisfeiler-Lehman (WL) is the fastest one, scaling very well on big datasets but at the expense of losing accuracy capabilities (Tables 4.3 and 4.7). Similarly, FB is the most efficient method in chemoinformatic graphs, being outperformed by (DGK) in social networks, but its classification performance is lower than our approach (Tables 4.4 and 4.5). Deep WL kernel remains a fast algorithm

Table 4.8: Runtime for features and kernels computations

Dataset	MaF	SP	DSP	GK	DGK	WL	DWL	FB
MUTAG	0.64 s	0.67 s	0.68 s	8.7 s	8.03 s	0.48 s	0.19 s	0.46 s
PTC	1.64 s	3.48 s	3.48 s	14.04 s	13.5 s	1.12 s	0.93 s	1.02 s
NCI1	26.34 s	45.78 s	50.11 s	2.8 min	2.7 min	17.36	13.20 s	13.01 s
NCI109	27.14 s	43.44 s	44.91 s	2.7 min	2.7 min	17.58 s	13.65 s	12.49 s
PROTEINS	9.12 s	39.75	40.29 s	46.8 s	42.7 s	5.21 s	16.9 s	5.93 s
ENZYMES	3.91 s	6.53	6.27 s	26.05 s	26.17 s	2.61 s	4.08 s	2.76 s
COLLAB	7.14 min	NA	NA	5.01 min	4.91 min	54.4 s	ME	29.31 min
IMDB-BINARY	8.88 s	NA	NA	46.4 s	44.60 s	2.46 s	2.05 s	6.7 min
IMDB-MULTI	5.59 s	NA	NA	48.12 s	46.7 s	2.37 s	1.91 s	7.18 min
REDDIT-BINARY	14.40 min	NA	NA	1.4 min	1.4 min	1.71 min	ME	2.67 min
REDDIT-MULTI-5K	29.55 min	NA	NA	3.8 min	3.7 min	5.98 min	ME	5.79 min
REDDIT-MULTI-12K	58.99 min	NA	NA	8.7 min	8.4 min	14.4 min	ME	10.93 min

in small size graphs, e.g., chemoinformatic datasets, but requires enormous memory (ME) for larger social networks, on which our method works well. Although our method shows longer run times on REDDIT datasets due to the eigenvectors computation performed on large networks, we still keep much better classification accuracies than other methods (Tables 4.3 and 4.4).

It is worth mentioning that the computation of multi-hop assortativities can be performed by means of Monte Carlo estimates on simulated random walks instead of the exact linear algebra formula (Eq. 4.3). This can be advantageous when the number of nodes and edges in the network, and the number of hops, preclude efficient matrix-vector computations. Note that in these circumstances, computing second, third, or other eigenvectors will also likely prove inconvenient. None of the benchmarks used in this article present these characteristics.

All our computations were done on a standard computer Intel(R) Core(TM) i7-4790 CPU, 3.60GHzI and 16G of RAM. The algorithms that exhausted our memory capabilities were flagged with ME (memory error) in the previous table.

Related methods

Graph classification has been broadly studied by the networks science and machine learning communities under different perspectives. Methods can be grouped in feature-based, and graph kernels approaches. We briefly introduce the methods against which we will compare our methodology.

There is a considerable amount of literature related to graph kernel-based methods. They can be categorized in three classes: graph kernels based on *random walks* [21], [98], kernels based on *subtree patterns*, [152], [162], [163], and also kernels based on *limited-size subgraphs* or *graphlets*, [164], [91]. The similarity between graphs is assessed by counting the number of common patterns or decomposing the input graphs into substructures, such as the shortest path or neighborhood subparts. However, kernels based on such substructures are computationally expensive, sometimes even NP-hard to determine and also limited in expressiveness. Moreover, the complexity of kernel computation for all pairs of networks in the training phase grows quadratically in the number of examples.

Improved graph kernels such as Weisfeiler-Lehman (WL) Subtree Kernel [162], and Neighborhood Subgraph Pairwise Distance Kernel [45] scale better by defining the similarity between a restricted, easy-to-compute class of substructures. A recent work [195] proposes a deep variant of the Graphlet, WL and Shortest-Path kernels. Patterns in the network are transformed into features in the same way that words are embedded in a Euclidean space in a natural language processing model, with the so-called CBOW / Skip-gram algorithms [129].

Automatic feature learning algorithms aim to learn the underlying graph patterns often through a neural network variant. More recently, [142] proposes to learn a convolution neural network (PSCN) for arbitrary graphs, by exploiting locally connected regions directly from the input graph. Shift aggregate extract network (SAEN) [144] introduces a neural network to learn graph representations, by decomposing the input graph hierarchically in compressed objects, exploiting symmetries and reducing memory usage.

The spectrum of graphs as features for graph classification has been investigated by [158], [190] who compute features from the spectral decomposition of the Laplacian matrix. Barnett [13] proposes a hybrid feature-based approach. It combines a manual selection of network features with existing automated classification methods.

Our method can be seen as a versatile feature-based model that creates multiscale patterns from any node attribute, whether they are structural, spectral, or exogenous (metadata), whether they are numerical or categorical. This is in contrast for example with Graphlet [164], Deep Graphlet (DGK) [195] or Feature-Based (FB) [13] methods who cannot exploit node metadata. The feature vector representation of a network is created using multi-hop assortativity

patterns from a random walker perspective. In this work, we show that even a minimal set of structural or spectral node attributes can be ‘amplified’ to a rich network feature representation allowing performant classification.

Conclusions and perspectives

In the present chapter, we introduced an extension of the intuitive notion of assortativity for networks and a particular application in graph classification. We define multi-hop assortativities by setting up a dynamic on the network and computing covariances over diverse node attributes among multiples time scales. It turns out that those features on a reduced number of structural attributes, in addition to node metadata whenever present, are useful to characterize networks. We use these assortativities in the context of network classification. The classification is performed by training a Support Vector Machine and Random Forest classifiers, achieving high accuracy on both social and biochemical datasets. The Friedman and Nemenyi post hoc tests were then used to determine whether the differences between the average ranked accuracies are statistically significant. The experimental results reveal that our approach is particularly effective when it is applied to large networks and datasets of many (possibly small) graphs, performing significantly better than kernel methods and convolutional neural networks. However, we show competitive accuracies when it is applied to small graphs with almost zero clustering coefficient, such as graphs representing molecules or proteins. When node metadata is available, our method outperforms kernel-methods and random walks based baselines and, remains competitive against neural networks approaches.

It is worth mentioning that although we experimented with networks with a single categorical node attribute, our approach applies to networks with scalar and multiple attributes, e.g., a social network where nodes attributes are age, weight, gender, or income of a person.

Our framework can also be tuned to incorporate the assortativity of any node attribute we deem relevant to the application, for instance, betweenness centrality, etc. Instead of the simple random walk, we may also use an application-specific diffusion dynamics, e.g., in continuous-time, biased towards some nodes, etc. [106,126]. In this proof-of-concept work, we only use the most straightforward dynamics, most elementary attributes with few eigenvectors, number of nodes, and edges.

An interesting application for future work is the classification of brain net-

works (connectome datasets) [10, 84]. Indeed, discriminating between healthy and non-healthy connectomes in various neural disorders is a task of growing importance in neuroscience.

In this chapter, we have introduced a new tool for network analysis employing random walks dynamics on networks. Generalizing the notion of assortativity allows us to capture higher-order patterns between node attributed at different scales. Also, we have proposed a hand-crafted approach to feature engineering on networks, useful to perform graph classification. In opposition to this chapter, the next chapter will present an unsupervised way to perform automatic feature learning from graphs.

5

Network embeddings with node identity awareness

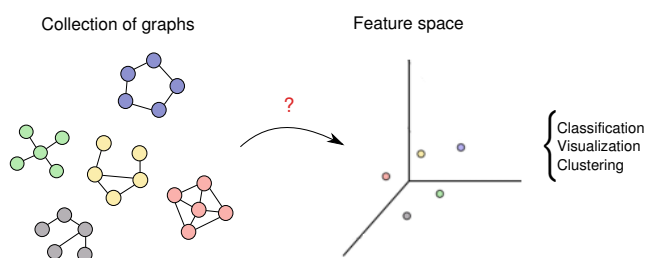


Figure 5.1: One graph mapped to one point in the feature space. Graph embedding methods seek to map input graphs into a lower-dimensional feature space, where similar graphs are close each other in the embedding space. The embedding representations are then suitable for downstream ML applications.

This chapter is about representation learning for networks. A central challenge in mining network-based data is finding effective ways to represent or encode graph structures so that it can be efficiently exploited by machine learning algorithms, see Figure 5.1. Several methods have focused on network representation at node/edge or substructure level. However, many real-life challenges related to time-varying, multilayer, chemical compounds, and brain networks involve analysis of a family of graphs instead of single one opening additional challenges in graph comparison and representation. Traditional approaches for learning representations rely on hand-crafted specialized features to extract meaningful information about the graphs, e.g., statistical properties, structural motifs, etc. as well as popular graph distances to quantify dissimilarity between networks.

Unlike the previous chapter, in this work we provide an unsupervised approach to learn graph embeddings for a collection of graphs defined on the same set of nodes so that it can be used in numerous graph mining tasks. By using an unsupervised neural network approach on input graphs, we aim to capture the underlying distribution of the data in order to discriminate between different networks. We assess our method empirically on synthetic and real-life datasets

and evaluate it in three different tasks: graph clustering, visualization, and classification. Results reveal that our method outperforms well-known graph distances and graph-kernels in clustering and classification tasks, being highly efficient in runtime.

Introduction

Numerous complex systems in social, medical, biological, and engineering sciences can be studied under the framework of networks. Network models are often analyzed at the node/edge or substructure level, studying the interaction among entities, identifying groups of nodes behaving similarly or finding global and local connectivity patterns among a given network. Furthermore, many real-life challenges might involve collections of networks representing instances of the system under study, e.g., functional brain networks (connectomes) [84], chemical compound graphs [174], multilayer networks [34], and so on. Other applications involve dynamic interactions between components, introducing additional complexity in the time evolution of the system. For example, in a mobile-phone social network, people are considered as nodes and the phone calls as edges. The dynamics of calls between users will systematically add and remove edges between them, describing a sequence of static graphs characterizing a dynamic evolution of the system.

With the increasing availability of manually labeled network data, many of these problems have recently raised the attention of the machine learning community. Machine learning applications seek to make predictions or discovering patterns in graph-structured data. For example, in chemoinformatics [48], one might need to predict the toxicity or, anti-cancer activity of proteins and molecules represented as graphs. In time-varying social networks, one might be interested in detecting unusual events [145], e.g., points in time in which the network connectivity differs abruptly concerning the evolution of the underlying process. Prediction of subjects having a neural disorder such as Alzheimer or Schizophrenia, based on their connectomes is crucial in neuroscience [75].

The cornerstone of this approach is the feature representation of the input data, e.g., finding effective ways to encode graph structures in such a way that it can be used in traditional machine learning models. For example, in order to predict whether a molecule is toxic or not, one might build a feature vector representation of a molecule incorporating information about its atoms, as well as global and local properties of the graph structure itself [13, 80]. By doing

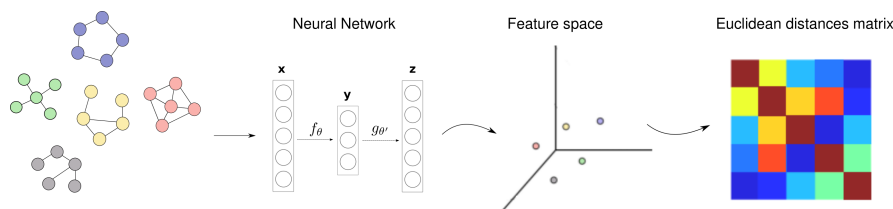


Figure 5.2: Overview of the proposed method. Given a family of graphs, we train an unsupervised neural network in order to uncover dissimilar relationships between graphs. The graphs are embedded into a feature space and mapped to a Euclidean distance matrix reflecting the structural similarity between input examples.

so, we can train a traditional machine learning model such as support vector machines, random forest, neural network, etc. so it will discriminate unseen toxic and non-toxic chemical compounds.

There exist many manners to extract features and comparing networks. For instance, graph distances [54, 119] such as the Jaccard and Hamming distances compute differences between graphs by counting the number of edit operations to transform a graph into another one, focusing mainly in their local connectivity patterns. Spectral distances rely on comparing the eigenvalues of the reference matrices representing the networks. Another popular class of distance measures is the graph kernels [163, 195]. A kernel can often be seen as the scalar product between implicit high-dimensional feature representations of a network [159]. The so-called kernel trick allows us to compare networks without ever computing explicitly the coordinates of data points in the high-dimensional feature space, sometimes with a substantial gain in computational time over classical graph-distance approaches. Moreover, some supervised approaches, e.g., requiring class labels for graphs, to learn embeddings of entire graphs have shown excellent performances in NLP tasks, i.e., semantic parsing and natural language generation *graph2seq* [193], graph classification with CNN for graphs [142] as well as metric learning approaches to brain network classification [123]. However, they pose some important limitations in practice, given the large volume of annotated data required to learn such deep neural networks architectures, as well as a limited expressiveness of the learned representations constrained to particular supervised tasks.

Because real-life networks are complex structures involving diverse connectivity patterns across domains, the expressiveness of the methods mentioned above is constrained when applied in multiple tasks. Therefore, the most relevant hand-crafted features tend to be task-dependent and often require special-

ized domain expertise in order to incorporate the right properties to perform accurately on the target task.

Unlike other approaches, in this chapter, we introduce a method to learn unsupervised network embeddings from a collection of networks, all defined on the same set of nodes. It should not be confused with node embedding approaches that aim to map nodes from a graph into vectors on a feature space (see [32, 71] for a survey of those methods). Therefore, this chapter refers to a graph or network embedding the outcome of mapping each network of a family as a vector in a Euclidean space (see Figure 5.2). The unsupervised nature of the method allows capturing the most relevant features from the data in order to produce a lower-dimensional representation of input graphs. This is crucial for overcoming the curse of dimensionality of high dimensional graphs. As a consequence, networks with similar structural properties will have neighboring embeddings in the feature space, and dissimilar graphs will be more distant. Our approach thus differs from the various definitions of graph distances or similarities mentioned earlier in that we learn automatically a feature representation of graphs assessing their similarity on a Euclidean space, instead of using a hand-crafted metric on the graphs. Besides, because many graphs coming from real-life applications rarely have exchangeable nodes, we focus on problems defined on networks that account for node identities, e.g., time-varying networks, brain networks, multilayer networks, etc.

Although the purpose of this work is not to make an extensive evaluation of graph clustering and classification tasks, we instead introduce and illustrate an unsupervised method for embedding networks on the same node-set, i.e., taking the node identity into account. We validate our method empirically in three network mining tasks: graph clustering (grouping similar graphs together), graph classification (predicting the class to which unseen networks belong to) and visualization (plotting many networks in $\mathbb{R}^2$). We perform diverse experiments on synthetic and real-life datasets such as time-varying networks (primary school network), multilayer networks (European airport network), and brain networks datasets.

This chapter is structured as follows. First, in Section 5.2 we will introduce some popular methods of the literature used to compare networks, as well as the development of the proposed approach. Subsequently in Section 5.3, we present some applications in graph visualization, clustering, and classification we performed on synthetic and real-life datasets. Subsequently, a computational analysis of our method is presented, finalizing in Section 5.4 with a

discussion and perspectives for future work.

Methods

Graph distances

Distinguishing among a class of networks requires a notion of distance or similarity between pairs of graphs [54]. These measures capture different aspects of the local and global structure of graphs, having an impact on the outcome of different applications. We present some of the most representative graph distances of the literature.

The *Hamming* and *Jaccard* distances are simple instances from the broader class of graph-edit distances [54]. They measure the number edge insertion and deletion operations necessary to transform one graph to another one. Denoting N the number of nodes of the undirected graphs $\mathcal{G}_1 = (\mathcal{V}, \mathcal{E}_1)$ and $\mathcal{G}_2 = (\mathcal{V}, \mathcal{E}_2)$ with adjacency matrices $\mathbf{A}_1$ and $\mathbf{A}_2$ respectively, the *Hamming* distance between them is defined as:

$$d_H(\mathcal{G}_1, \mathcal{G}_2) = \frac{1}{N(N-1)} \sum_{i,j}^N |\mathbf{A}_1 - \mathbf{A}_2|_{i,j} \quad (5.1)$$

which defines a scaled version of the $L_{1,1}$ norm between matrices bounded between 0 and 1. Similarly, the *Jaccard* distance is defined as:

$$d_J(\mathcal{G}_1, \mathcal{G}_2) = \frac{|\mathcal{E}_1 \cup \mathcal{E}_2| - |\mathcal{E}_1 \cap \mathcal{E}_2|}{|\mathcal{E}_1 \cup \mathcal{E}_2|} \quad (5.2)$$

where $\mathcal{E}_1$ and $\mathcal{E}_2$ are the set of edges for the graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ respectively.

DeltaCon [103] is a popular graph similarity measure in connectomics. As the edit distances it also exploits node correspondence across graphs. The intuition behind the method is to compute first pairwise node similarities of input graphs through a variant of a personalized PageRank algorithm [103]. The pairwise node affinity matrices $(\mathbf{S}_1, \mathbf{S}_2)$ are compared using the Matusita Distance defined by:

$$d_{DC}(\mathbf{S}_1, \mathbf{S}_2) = \sqrt{\sum_{i,j=1}^n (\sqrt{\mathbf{S}_1(i,j)} - \sqrt{\mathbf{S}_2(i,j)})^2} \quad (5.3)$$

On the other hand, the *spectral distances* for graphs have proven to be very useful in many applications [125, 191]. However, the spectral nature of the method makes it invariant to node permutations. Roughly speaking, these methods compare the spectrum of any matrix representing the input graph, generally the graph Laplacian. The combinatorial Laplacian matrix (CL) of an undirected graph $\mathcal{G}$ is defined by $\mathcal{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{D}$ is the diagonal matrix whose i -th element equal to the degree of node i , and $\mathbf{A}$ its adjacency matrix. The normalized Laplacian matrix (NL) is defined by $\mathcal{L}' = \mathbf{D}^{-1/2} \mathcal{L} \mathbf{D}^{1/2} = \mathbf{I} - \mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{1/2}$, with $\mathbf{I}$ the corresponding identity matrix. We denote the eigenvalues of any of the Laplacian matrices as $0 = \lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_N$.

For any $\mathcal{L}$ and $\mathcal{L}'$ we consider the following spectral distance. The spectral distance between two undirected graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ is defined as [125]:

$$d(\mathcal{G}_1, \mathcal{G}_2) = \sqrt{\sum_{i=1}^{n_\lambda} [\lambda_{N+1-i}(\mathcal{G}_1) - \lambda_{N+1-i}(\mathcal{G}_2)]^2} \quad (5.4)$$

where n_λ is the number of eigenvalues considered for the computation of the distance (typically $n_\lambda = N$).

Embedding distances

Unlike the previous distances, our approach performs network comparisons directly on a feature space through a learned non-linear mapping applied to input graphs (see Figure 5.2). We will explain the building blocks of our method in the following subsections.

Denoising Autoencoder (DAE)

Unsupervised learning approaches aim to uncover hidden patterns or learning representations from unlabeled data. As was presented in Section 2.2.3, the classical Autoencoder aims to minimize the reconstruction error of the input with it self. Minimizing such reconstruction criterion alone is unable in general to guarantee the extraction of meaningful features as it can potentially memorize the training data. We want the autoencoder to be sensitive enough to recreate the original observation but insensitive enough to the training data such that the model learns a generalizable encoding and decoding mapping.

In order to overcome this limitation, the objective function (Eq. 2.34) is redefined in such a way that the autoencoder will be able to clean partially corrupted input or simply denoising it. This modification leads to a simple

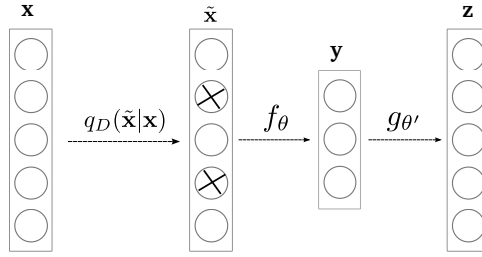


Figure 5.3: Denoising Autoencoder. A corrupted instance $\tilde{\mathbf{x}}$ of a graph $\mathbf{x}$ is fitted into the Autoencoder’s input. The Autoencoder is trained to recover a cleaned version of the input by compressing it through a non-linear mapping f_θ and mapping it back (through $g_{\theta'}$) to a reconstructed version of the original input graph $\mathbf{x}$.

variant of the basic autoencoder described above. A denoising autoencoder (DAE) [183] is trained to reconstruct a clean or repaired version from a corrupted input. This is done by transforming the original input $\mathbf{x}$ in $\tilde{\mathbf{x}}$ through a stochastic mapping $\tilde{\mathbf{x}} \sim q_D(\tilde{\mathbf{x}}|\mathbf{x})$. By doing so, the AE is forced to learn meaningful features, robust under the corruption of the input.

The corrupted version $\tilde{\mathbf{x}}$ is mapped with the original autoencoder to a hidden representation $\mathbf{y} = f_\theta(\tilde{\mathbf{x}})$ from which we reconstruct a clean $\mathbf{z} = g_{\theta'}(\mathbf{y})$. An important observation is that $\mathbf{z}$ is now a deterministic function of $\tilde{\mathbf{x}}$ rather than $\mathbf{x}$. See Figure 5.3 for an schematic representation of the model. Thus, we optimize the same objective than Eq. 2.34 but replacing $\mathbf{x}$ by $\tilde{\mathbf{x}}$. Optimization is done with the standard mini-batch gradient descent and back propagation algorithms [109].

Network embedding distances

The adjacency matrix $\mathbf{A}$ of a graph is a simple network representation but alone can be insufficient as an input for the DAE. It only captures first-order relationships between neighboring nodes. We extend this by computing higher powers of the adjacency matrix in order to capture multiple paths relationships. Thus, we consider $\mathbf{A}^r$ for some $r \geq 1$ as a more adequated input for the Denoising Autoencoder.

Note that as the class of problems we tackle is defined on a collection of networks having a node correspondence across graphs, our method remains invariant to the node order when the same node permutation is assigned to the graphs.

The vectorization of matrices is required to feed the graphs into the DAE

input. Let $\mathbf{A}^r$ the r power of the $n \times n$ adjacency matrix $\mathbf{A}$ of a graph. The vectorization of $\mathbf{A}^r$ is a $n^2 \times 1$ column vector $\mathbf{x} = \text{vec}(\mathbf{A}^r)$ obtained by staking the columns of $\mathbf{A}^r$. Notice that when the graph is undirected, the input matrix can be described with a $\frac{n(n+1)}{2} \times 1$ column vector $\mathbf{x}$. We apply a stochastic noise on the input by removing or adding a small fraction of edges at random, and then we infer the parameters of the DAE using the noisy inputs $\tilde{\mathbf{x}}$ as was presented in the previous section.

The optimal solution $\theta^* = \{\mathbf{W}^*, \mathbf{b}^*\}$ parametrizes an encoder mapping f_{θ^*} of the DAE. It embeds the input $\mathbf{x} = \text{vec}(\mathbf{A}^r)$ into a smaller dimensional vector $f_{\theta^*}(\mathbf{x}) \in \mathbb{R}^d$. A main advantage of transforming graphs into feature vectors is that it allows us to compare easily networks computing only Euclidean distances between their embeddings. Hence, the *network embedding distance* between two graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ with power matrices $\mathbf{A}_1^r$ and $\mathbf{A}_2^r$ is defined as:

$$d_{Emb}(\mathcal{G}_1, \mathcal{G}_2) = \|f_{\theta^*}(\text{vec}(\mathbf{A}_1^r)) - f_{\theta^*}(\text{vec}(\mathbf{A}_2^r))\|_2. \quad (5.5)$$

In the following we introduce other related work on graph embeddings.

Other graph embeddings

Inspired by the success of the recent document embedding models [108], *graph2vec* [134] extends the same ideas in the graph domain. This method proposes to consider a graph as a document where rooted subgraphs around every node in the graph are seen as words composing the document. The underlying assumption is that different subgraphs compose graphs in a similar way that different words compose sentences/documents. The graph embedding is then trained following the popular skip-gram model [129], sampling rooted subgraphs around different nodes as input vocabulary. These non-linear substructures allow capturing structural equivalence patterns ensuring that *graph2vec* representation learning will yield similar embeddings for structurally similar graphs. Similar to this, *subgraph2vec* [133] is an unsupervised method to learn subgraph embeddings but sampling linear substructures such as fixed-length random walks.

On the other hand, Graph Attention Model (*GAM*) [112] is an end-to-end approach for graph classification based on Recurrent Neural Networks (RNN). The attention mechanism allows the method to process a portion of the graph, avoiding noise in the rest of the graph. The idea is to sample some nodes with a random walk endowed with an attention mechanism guiding the walk. It helps to biases the walks towards neighboring nodes having similar node labels, exploiting local graph information. In the end, the RNN component will

integrate information gathered from different parts of the graph, accumulating the historical representation of the data for predicting the output graph label. In practice, the history layer h_t is taken as graph embedding for our experiments.

It is worth mentioning that we introduce and illustrate an unsupervised method for embedding networks on the same node-set, i.e., taking the node identity into account. Although we found no exact competitors in the literature, making systematic comparison difficult, it is illuminating to compare it numerically on some typical tasks with edit-distances like *Hamming's* and *Jaccard's* (which are node-id aware but not learning an embedding from a given family of networks), *DeltaCon* which is also node-id aware but not learning an embedding, spectral methods (which are not node-id aware, and not learned), *Graph2Vec* and *Subgraph2vec* (which learn unsupervised embeddings but are not node-id aware) and *GAN*.

In the following sections, we present some experimental results of our method in various synthetic and real-life applications.

Experiments and Results

The experiments have three purposes. First, they assess the performance of our method in discriminating different types of networks which are generated from different models, edge densities, and heterogeneous community structure. Next, they show the use of graph embeddings in networks coming from diverse real-life applications such as time-varying networks, connectomes, and multilayer networks. Finally, they highlight the runtime performance of feature computation and compare it against other techniques.

It is worth mentioning that all our experiments were executed with $\mathbf{A}^3$ where $\mathbf{A}$ is the adjacency matrix of the graph, as input for the *DAE*. As $\mathbf{A}^3$ counts the number of 3-hop walks between pairs of nodes, it will encode a 'mesoscopic' view of the graph beyond immediate neighbor-neighbor relations. For instance, it includes on the diagonal the number of triangles; a feature deemed representative of the clustering of the complex networks [138].

An essential challenge when working with embedded data is to decide the embedding dimensionality. Higher embeddings preserve more information while are more expensive in computation than lower embeddings. Besides, different embedding methods beyond graph embeddings, propose their own right dimension, such as Node2vec with 128 dim [79], Word2vec with 300 dim [129],

Table 5.1: Summary of synthetic datasets

DATASET	Type of network	Properties	True clusters
ER	Erdős-Rényi	Different average degrees	3
Mixed	ER - Power law	Different models, same average degree	2
LFR	Power law	Strong vs weak communities strength	2
Dynamic	Erdős-Rényi	Perturbation mechanism: rewiring, adding and removing a % edges	3

spectral clustering for social networks classification with 300, 500, 600 [177] to give some examples. Thus, the dimension of our graph embedding was determined empirically and fixed in all experiments to 800. The dimension for the considered graph embedding methods was also 800. They were set as the values providing the best clustering performance. The training hyper-parameters for the *DAE*, *graph2vec*, *GAM* and *subgraph2vec* methods can be found in the appendix section.

We evaluate our approach on three different but related tasks: graph visualization, graph clustering, and classification.

Synthetic datasets

We generate four synthetic datasets incorporating different structures, degree distributions, and edge densities. Each dataset has 600 networks with 81 nodes. An overview of the generated datasets is shown in Table 5.1. Their properties are detailed in the following:

- In the first synthetic dataset, we generate three Erdős-Rényi networks (*ER*) with different parameters, producing random networks with different average degrees. Then we make 200 copies from each graph reordering the nodes with a different permutation of the original graph.
- In the second dataset named *Mixed*, we generate two groups of networks: a collection of power-law networks generated with the Barabási-Albert (*BA*) model and Erdős-Rényi networks, all of them with the same average degree of 6.

- In the third dataset, power-law networks were generated using the Lancichinetti–Fortunato–Radicchi (*LFR*) benchmark [61]. This algorithm creates networks with heterogeneous structures and community sizes. The mixing parameter $\mu \in [0, 1]$ controls the strength of the community arrangements, achieving well-defined communities with small μ , meaningless community structure when μ is close to one and $\mu = 0.5$ as the border beyond which communities are no longer defined in the strong sense [151]. Thus, we generate two groups of networks: with mixing parameter $\mu = 0.1$ and $\mu = 0.5$. Other parameters are common for both groups: number of nodes $N = 81$, average node degree equal 11, community sizes varying between 6 and 22 nodes, exponent for the degree sequence 2 and exponent for the community size distribution 1. Therefore, the two groups of networks differ only in the strength of their communities structure and not in the degree distribution, being a more challenging problem than the previous dataset.
- In an attempt to simulate a dynamic network evolution, we simulate a time-varying network (*Dynamic*) following [54], applying a perturbation mechanism from a starting ER network. At each time step, a fraction of the edges of the previous graph are rewired uniformly at random. At the same time, we apply a depletion/thickening process in which edges are deleted with probability 0.015 and formerly absent edges are added with probability 0.015. We introduce two perturbation points by augmenting the probabilities of adding and deleting edges to 0.2 from time $t = 200$ and also to 0.6 from time $t = 400$, defining three ground-truth clusters of similar behaving networks.

For all datasets, we generate balanced ground truth classes.

Applications in graph visualization

A useful application of network embedding is graph visualization . It mainly consists of representing graphs as 2D points, e.g., an entire graph as one point, maximizing a certain notion of similarity. Considerable research has been done in visualizing nodes of graphs based on the premise that nodes sharing common structures, e.g., neighboring nodes, structural equivalent nodes, assortative nodes, etc. should be mapped to close points in the embedding space [32, 71].

In contrast, we propose to visualize multiple graphs at once on a two-dimensional space in the following way. From a given family of graphs, their

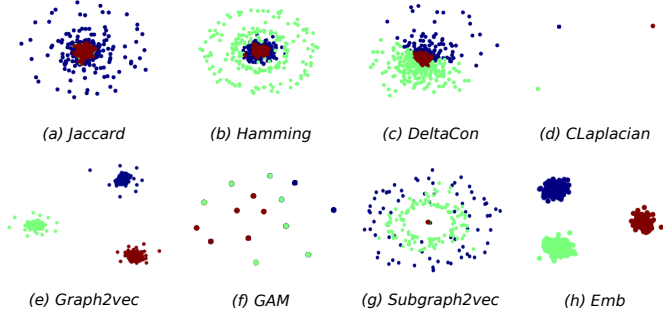


Figure 5.4: Visualization of permuted Erdős-Rényi (ER) dataset. Each point corresponds to a network. Color of a point indicates the category of the network according with its average degree $\langle d \rangle$. For blue $\langle d \rangle = 4$, green $\langle d \rangle = 6$ and red color $\langle d \rangle = 8$.

embeddings are learned and used to compute the embedding distance matrix (Eq. 5.5). The same procedure is applied for the graph embeddings described previously in Section 5.2.2. In order to enable visualization, a methodology is needed to bring the embedding distances into a low-dimensional visualization. We choose the Multi-scale SNE tool [111] as a standard method. This is a non-linear dimensionality reduction approach for data visualization which aims to reproduce in a low-dimensional space the local and global neighborhood similarities observed on any similarity matrix. Multi-scale SNE can be seen as an improvement of SNE based methods such as t -SNE [181], in that it actually optimizes the choice of bandwidths similarities providing a parameter-free method for data visualization. In this way, we expect that networks with similar properties as learned by the Denoising Autoencoder are neighboring points in the two-dimensional visualization, while the gap between different groups of graphs is maximized.

We perform visualization of some datasets from Table 5.1. As networks within a particular dataset were generated with a range of parameters, we assign a color to each point in the visualization, reflecting the value of the parameter used to generate the network. In this way, we expect that a good visualization will preserve the same colored points as neighbors points in $\mathbb{R}^2$ maximizing the gap between groups.

Discussion

Figure 5.4 shows the visualization of the *ER* dataset after applying our method together with the methods introduced in the aforementioned sections. As can

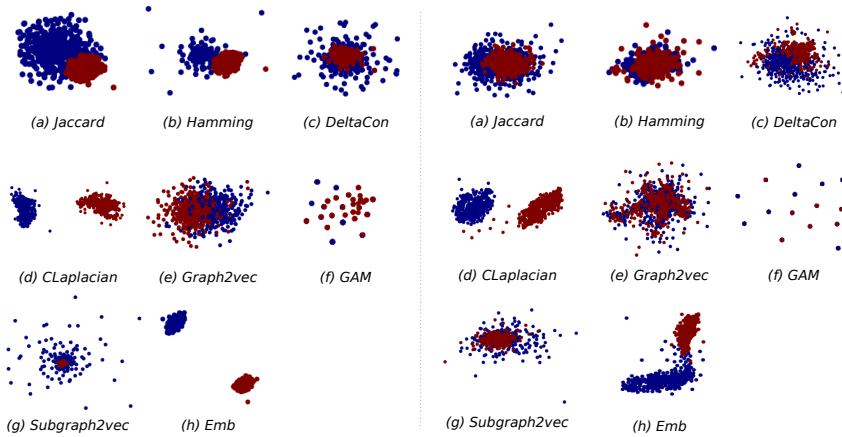


Figure 5.5: Visualization of networks when applying multi-scale SNE on the distance matrix defined by graph distances and embeddings methods. (Left) networks from *Mixed* dataset. Blue color correspond to *ER* networks and red color are *BA* networks, both groups with $\langle d \rangle = 6$. (Right) Networks generated with the *LFR* benchmark. Blue color correspond to networks with well defined communities, $\mu = 0.1$ whereas red color are networks generated with $\mu = 0.5$

be expected, results with *Jaccard* and *Hamming* distances are not satisfactory because points from different groups overlap. Even though *DeltaCon* tries to separate the data, the boundary between groups was not clearly determined. Because spectral distances are permutation invariant measures, they collapse all permuted graphs to the same point showing a hard separation between classes. Regarding graph embeddings, *graph2vec* makes a good job grouping the data in three non-overlapping clusters, whereas *subgraph2vec* creates a concentric cloud of points collapsing different graphs to the center. On the other hand, *GAM* captures a different notion of similarity, where graphs with locally similar nodes degree sequences are grouped. Finally, our embedding (*Emb*) shows three well-defined clouds of points grouping together isomorphic graphs. Our method exploits node correspondence across graphs when it is known, but even if the node order is not of particular significance, we can retrieve essentially identical networks.

Similarly, in Figure 5.5 we show the visualizations of *Mixed* and *LFR* datasets. We can make the following observations: *CLaplacian* and our graph embedding method (*Emb*) are capable to discover the differences between data examples, so that they separate the data almost perfect clusters. Although edit distances (*Jaccard*, *Hamming*) work better in discriminating networks with distinct degree distribution than networks with community structure (*LFR*),

their discriminative power is still poor. *GAM* as in Figure 5.4 collapses networks with local similarities into the same embedding, performing poorly in this benchmark. Even though *subgraph2vec* does not capture perfectly the differences in *ER* and *Mixed* datasets, it performs better than *graph2vec* on networks with community structure.

Finally, Figure 5.6 shows the visualization of the *LFR* dataset in more detail. The left-hand side plot shows two clouds of points encoding networks with different mesoscopic structures. As can be seen, the blue cluster tends to spread more than the red one, which is more compact. This illustrates the structural variability of networks having heterogeneous number/size communities (blue cluster) against a group of networks with weakly modularity (red cluster).

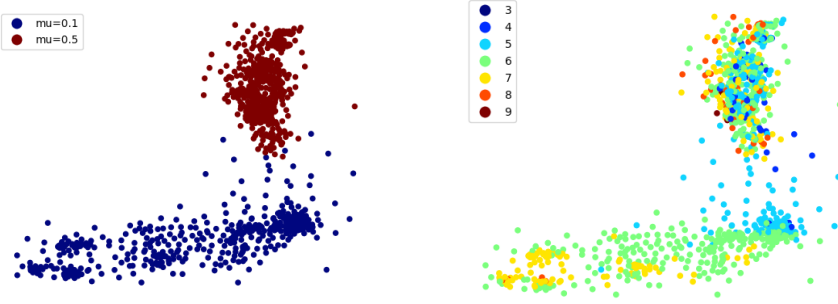


Figure 5.6: Visualization of networks when applying multi-scale SNE on the distance matrix defined by our embeddings on the LFR benchmark. (Left) networks with different community strength: $\mu = 0.1$ and $\mu = 0.5$. (Right) Same left-hand side networks. Colors encode the number of planted communities within each network.

In the right-hand side plot of Figure 5.6, we keep the same networks from the left-hand side plot, but we color them according to the number of ground-truth communities on each network. Inspecting the bottom cluster, we observe that even if there is not a clear grouping of points, the data is distributed in a quasi-continuum manner, having networks with a similar number of communities as neighboring points in the plane. On the other hand, the networks on the top are blended, which is expected as they have a weak community strength. This visualization allows us to understand the notion of similarity captured by the Autoencoder on the underlying dataset.

Once more, we emphasize that although the embedding is in principle dependent on the order of the nodes, in this specific case, different orderings lead to closely similar visualizations. This is foreseen as in this case, albeit all the networks are supported on the same number of nodes, there is no natural one-

to-one correspondence between the nodes of two networks, and all nodes are treated symmetrically in the generation process.

Visualizing real-life networks: temporal networks

The primary school network [175] is a dataset containing temporal face-to-face interactions between 232 children and ten teachers in a primary school in Lyon, France. The data was collected over two days (Thursday, October 1 and Friday, October 2, 2009) spanning from 8:45 am to 5:20 pm the first day, and 8:30 am to 5:05 pm the second day.

The dynamic evolution of the network can be modeled as a time-varying network defined on a fixed number of nodes, and dynamic edges representing the physical interaction between children and teachers. It can be represented as a sequence of static graph snapshots over a time window τ , which aggregates all events or edge activations that occurred between the interval $[(t-1)\tau, t\tau]$. For this experiment, we chose a time resolution $\tau = 20s$ yielding 1555 snapshots for Thursday, 01-October. Its visualization is shown in Figure 5.7.

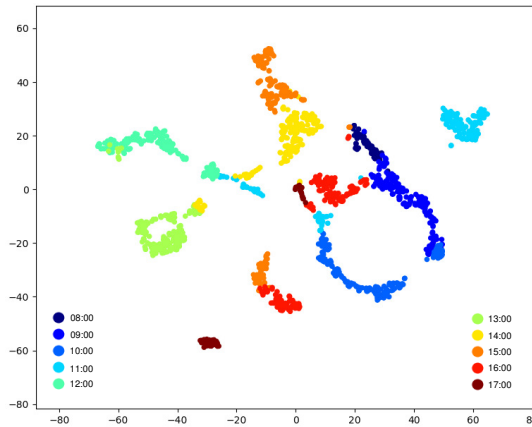


Figure 5.7: Visualization of networks when applying multi-scale SNE on the distance matrix defined by our embeddings on the primary school network. Each point represents a network snapshot during the day of the 01-October-2009. Color of points encode a time frame of the day spanning from 8:45 until 17:20

The clusters in Figure 5.7 can be interpreted as groups of networks behaving similarly and correlated with external events, e.g., consecutive clusters are separated because of an external event. For instance, lunchtime is characterized by clusters between 12:00 and 14:00. The class time is represented by a long cluster of dark and light blue points in the morning and yellow, orange groups

in the afternoon. The end of the school day is highlighted with a brown group of points. Mixed group colors indicate smooth temporal transitions, e.g., end of lunchtime and beginning of classes (green-yellow-light blue), also the end of the afternoon break to classes (orange-red). Therefore, we discover that events are reflected in the embedding space as they can be identified by clusters of similar networks. However, we stress that drawing conclusions about the meaning of progressive lines or compact clusters should be taken with care, as they may be an artifact of the visualization. Thus, this states an interesting direction for further investigation.

Note that unlike synthetic examples from the previous section, nodes have an individual identity, and different network snapshots take place on the same set of nodes.

Applications in graph clustering

Clustering synthetic graphs

Another important application is the clustering of networks. Clustering aims to group a set of networks, where networks within the same group (cluster) are more “similar” to each other than those in separated clusters. We proceed similarly to the previous section, but we do not perform dimensionality reduction to $\mathbb{R}^2$. Instead, clustering is performed directly in the embedding space with the standard spectral clustering algorithm [141]. This technique apply clustering to a projection of the normalized Laplacian defined in terms of any similarity matrix of the data.

We run our method on each dataset from Figure 5.1 and compute a 600×600 network embedding matrix using Eq. 5.5. In order to compare against other techniques, the graph distance matrices for the methods introduced in Section 5.2.1 and 5.2.2 are computed. All matrices are normalized having a maximum entry of one for similar pairs of graphs and zero for the most dissimilar ones. Therefore, spectral clustering is performed on the similarity matrices induced by graph distances and other graph embeddings.

The clustering performance is evaluated through the normalized mutual information (NMI) [31] metric in the form:

$$NMI(C_t, C_p) = \frac{2I(C_t; C_p)}{H(C_t) + H(C_p)} \quad (5.6)$$

where H is the entropy of a class distribution and I the mutual information between the ground truth class distribution C_t , and the predicted cluster as-

segment C_p . It runs from zero when the algorithm fails to a value of one when the clustering is perfectly recovered. Details about the datasets are presented in Table 5.1.

In order to evaluate the sensitivity of the clustering to the node ordering, we perform clustering with different enumeration of nodes by applying a fixed node permutation across the networks. We reported the mean and standard deviation of the NMI after running the experiment ten times.

Table 5.2: Graph distances. Clustering results for synthetic datasets (NMI).

	Hamming	Jaccard	DeltaCon	CLP	CLP normed	Emb
ER	0.239	0.603	0.581	1.0	1.0	1.0 $\pm$ 0.0
Mixed	1.0	1.0	1.0	1.0	0.364	1.0 $\pm$ 0
LFR	0.269	0.284	0.278	0.766	0.973	0.995 $\pm$ 0.005
Dynamic	0.389	0.232	0.205	0.237	0.195	0.452 $\pm$ 0.085

Table 5.3: Graph embeddings. Clustering results for synthetic datasets (NMI).

	Graph2vec	GAM	Subgraph2vec	Emb
ER	1.0	0.11	1.0	1.0 $\pm$ 0.0
Mixed	0.432	0.004	0.983	1.0 $\pm$ 0
LFR	0.002	0.0001	0.009	0.995 $\pm$ 0.005
Dynamic	0.074	0.001	0.219	0.452 $\pm$ 0.085

Discussion

Regarding the clustering results in Table 5.2, we observe that our graph embedding (*Emb*) provides better clustering than traditional graph distances. The method is capable of differentiating networks with different edge densities (*ER*). Meanwhile, it is also able to discriminate networks with different degree distributions even if they have a similar average degree (*Mixed*). Discriminating power-law networks from strong to weak community structure (*LFR*) is also well achieved. The time-evolving network (Dynamic) is a harder setting in which our method performs the best comparatively to graph distances. In this case, the graph embeddings can capture the variations introduced by change points in the underlying evolution of the network. This can be explained because the DAE was not designed for a target sort of graphs. Instead, it learns the underlying distribution of the data, identifying the main factor of variability adapting its parameters for discriminating networks with different structures.

The quality of the embeddings remains unalterable after node permutations, which is confirmed by the low variance in the NMI. Hence, in practice, we fixed a node ordering for the learning procedure.

In addition, as can be seen in Table 5.3 all graph embedding methods have a good performance on the *ER* dataset except *GAM*. We found that globally *subgraph2vec* performs better than *graph2vec*, but our embedding approach (*Emb*) performs the best.

Clustering real-life networks: multilayer networks

The European Air Transportation Network (ATN) [34] is a multilayer network with 37 layers, each representing a different European airline. Each layer has the same number of nodes representing 450 European airports. We learn graph embeddings for every layer, and we cluster them by applying a standard hierarchical clustering algorithm on the network embedding distance matrix. The hierarchical clustering provides a partition of layers according to their similarity on the embedding space, see Figure 5.8.

Our findings confirm those introduced in [34]. We can identify two main clusters representing major and low-cost aerial companies, as well as some regional airlines arranged together. Indeed, these airlines have developed according to different structural/commercial constraints. Low-cost companies tend to avoid being centralized and cover more than one country simultaneously. Major airlines have a hub and spoke network, connecting outlying airports to few central ones, providing maximum coverage from their home country.

Applications in graph Classification

We evaluate graph classification in the context of supervised classification. It requires previously annotated reference samples (graphs) in order to train a classifier and subsequently classify unknown data.

Brain connectomes classification

In this experiment, we apply our method on brain networks (connectomes) built from a magnetic resonance imaging (MRI) [10] dataset. Structural and diffusion MRI data of 91 healthy men and 113 healthy women is preprocessed in order to create undirected networks. All graphs have the same 84 nodes representing neural Regions of Interest (ROIs). Weighted edges correspond to the number of neural fibers linking two ROIs. The ROI keeps the same

correspondence among graphs. The task is to classify connectomes according to gender, male or female.

Experimental setup

We assess the performance of our method against some well-known algorithms for graph classification, mainly graph kernels, and feature-based approaches. We choose the Shortest Path (SP) and the Weisfeiler-Lehman (WL) subtree kernels [163]. We also compare against the feature-based (FB) method [13] and Multi-hop assortativities features (MaF) for network classification [80]. Such methods provide a pairwise similarity matrix between networks in the form of a Gram matrix which is used to train a popular support vector machine classifier (SVM) [23]. Note that the graph distances considered in this work do not define a proper positive semi-definite matrix. Therefore, following [192] we shift the spectrum of their similarity matrices providing a proper kernel coherent with the SVM setting.

We follow the experimental setup of [163,195]. The dataset is split randomly in training and testing sets. The best model is cross-validated over ten folds. Parameters of SVM are optimized only on the training set. Thus, we compute the generalization accuracy on the unseen test set. In order to exclude the random effect of the data splitting, we repeated the whole experiment 10 times. Finally, we report the average prediction accuracies and its standard deviation.

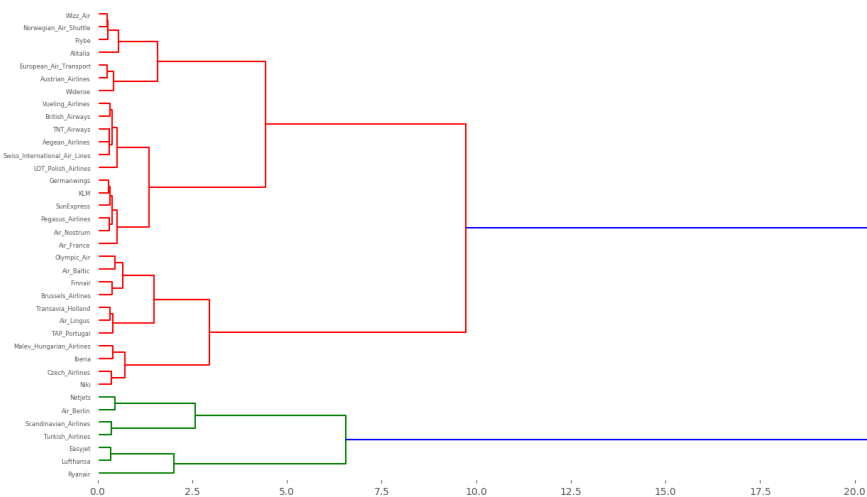


Figure 5.8: Dendrogram of airlines for European airports

For each graph kernel, we report the result for the parameter that gives the best classification accuracy. For the feature-based approach [13], feature vectors were built with the same network features they reported in their paper: number of nodes, number of edges, average degree, degree assortativity, number of triangles and global clustering coefficient. Results are shown in Table 5.4 and Table 5.5.

Table 5.4: Mean and standard deviation of classification accuracies on brain connectomes dataset.

WL	SP	FB	CLaplacian	NLaplacian	Emb
61.20 ± 2.16	65.45 ± 1.78	65.95 ± 2.54	74.19 ± 11.16	71.07 ± 10.95	87.20 ± 7.60

Table 5.5: Mean and standard deviation of classification accuracies on brain connectomes dataset.

Hamming	Jaccard	DeltaCon	MaF	Emb
84.37 ± 9.26	84.34 ± 10.11	87.80 ± 6.54	84.26 ± 5.81	87.20 ± 7.60

Discussion

As can be seen in Table 5.4, WL, SP, and FB perform significantly worse than spectral distances and graph embedding. This is expected as they do not take the identity of the nodes into account. Here, all brains share the same anatomical regions, which make the order of the nodes relevant. In Table 5.5 can be seen that among the approaches exploiting node correspondence, our method (Emb) remains competitive given the high variation of the methods across different folds.

Computational cost

Our graph embedding approach involves globally two steps: learning graph embedding through the DAE followed by a pairwise Euclidean distance matrix computation (Eq. 5.5). For the considered methods listed in Table 5.2 (except *Emb*), we measure the running time for deriving the pairwise distance matrix. For the graph embedding approaches from Table 5.3 including our approach (*Emb*), we report the running time for the entire process, including learning the graph embeddings and computing the Euclidean distances. Results are

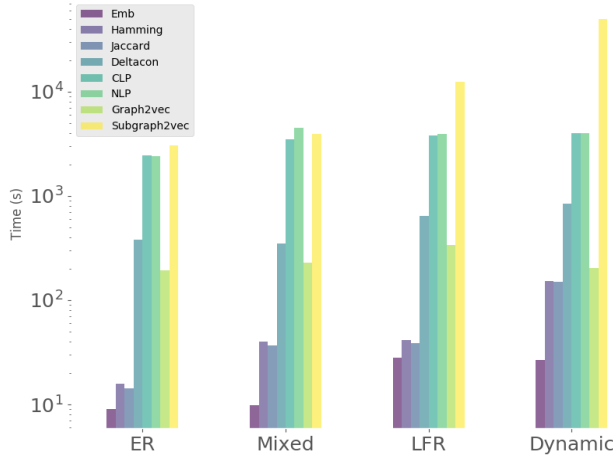


Figure 5.9: Computational time for feature computation. Time is log scaled.

shown below in Figure 5.9. As can be seen, our method (*Emb*) outperform all graph distances across all studied datasets. *Hamming* and *Jaccard* distances are globally fast methods because they rely on simple edge-level computations being slower in dense networks. It is known that spectral distances (*CLP*, *NLP*) are heavy in computation due to the entire eigenvalue calculation. Even if *Deltacon* is a scalable graph similarity measure, it is outperformed by the edit distances (*Hamming*, *Jaccard*), but prevails over spectral distances. On the other hand, *graph2vec* outperforms the hand-crafted spectral and DeltaCon methods being efficient for a method without node identity awareness. *Sub-graph2vec* is globally the slowest method, showing that modeling varying size context for the skip-gram model is considerably more expensive than fixed-size neighborhoods. Meanwhile, our graph embedding method remains the fastest. Indeed, the performance of mini-batch gradient descent is proportional to the batch size; thus, it converges faster on relatively small datasets. For efficiency reasons, the Euclidean distance between two feature embeddings x, y was computed as $d(x, y) = \sqrt{\langle x, x \rangle - 2\langle x, y \rangle + \langle y, y \rangle}$. This formulation has the advantage of being very efficient for sparse graphs given that some terms can be pre-computed for an entire pairwise computation.

All computations were done on a standard computer Intel(R) Core(TM) i7-4790 CPU, 3.60GHzI with 16G of RAM.

Conclusions and perspectives

In the present chapter, we proposed a method to learn graph embeddings for a collection of networks defined on the same set of nodes, e.g., mapping graphs to $\mathbb{R}^p$ vectors. Our method allows comparing graphs computing only Euclidean distances between their embeddings in a feature space. We evaluate our method in three different applications in graph clustering, visualization, and classification. Across various synthetic and real-life datasets, we compare our approach against well-known graph distances, graph-kernel methods, and unsupervised graph-embedding methods of the literature.

It turns out that our approach extracts the most appropriated features to distinguish different kinds of graphs. Indeed, clustering collections of similar networks yield good quality partitions among synthetic datasets (Table 5.2), discriminating better heterogeneous structures among networks. Although there is not an explicit agreement about the use of combinatorial or normalized Laplacian in graph mining applications, spectral distances are highly competitive in graph clustering and visualization but are incapable of exploiting node correspondence. Nevertheless, our learned graph embeddings turn out to be computationally cheaper than the considered methods (Figure 5.9), being an attractive yet efficient method for comparing networks with equal size.

The results in graph classification reveal that our approach has superior performance than graph-kernels and graph spectral distances (Table 5.4). Indeed, exploiting the node identities across graphs increases the accuracy of the method. Thus, this result suggests a promising research direction in the connectomics field.

Note that in this work, we were not focusing on the task of, for instance, distinguishing random networks with different average degrees, which can be trivially solved without any machine learning tool. Instead, we aimed to show an automatic way to leave the machine to figure out the most relevant hidden patterns from the data, which is more general than designing tailored methods for particular applications.

One limitation of the current study is the assumption that all networks must have the same set of nodes. Even if in many real applications this hypothesis holds, a large number of complex systems have varying size graphs, e.g., chemical compounds, social networks, etc. This study has only investigated the class of graphs without node/edge attributes, such as age, gender in social networks. Besides, regarding the scalability on large networks, addressing these issues

introduce additional challenges and new opportunities for further research.

Despite this limitation, our work has the potential of being extended in two directions. Because the DAE captures the underlying probability distribution of the data [183], the decoding function could be used to generate artificial data, e.g., generating brain networks, for mining purposes. Another possibility is to explore deeper neural network architectures such as the stacked autoencoders [184] and its variants in order to learn hierarchical feature representation of the data for downstream graph applications.

The next chapter will introduce stability of biomarkers selection for prediction of schizophrenia in brain networks.

Schizophrenia diagnosis in the human connectome

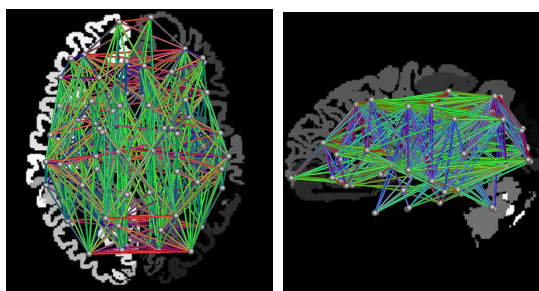


Figure 6.1: **Structural connectome**: an abstract representation of the structure of the human brain as a complex network. Gray spheres represent the ROIs. The color of the edges corresponds to the orientation. Left: axial view. Right: sagittal view. Taken from [10].

The human brain is composed by approximately 85 billions of neurons, each of them generating thousands of synaptic interactions. Thanks to the recent progress in neuroimaging, we are currently able to explore the large-scale structure of the human brain.

The connectome framework allows representing the brain as a complex network (Figure 6.1). The brain connectivity network can be expressed in terms of structural, functional, and diffusion MRI. They describe either anatomic interactions between brain regions of interest (ROIs) through their fiber density (Structural Connectome, SC) or mapping functional correlations between fMRI time courses (Functional Connectome, FC). The identification of such biomarkers in schizophrenia could lead to clinically useful tools for establishing both diagnosis and prognosis. From a machine learning perspective, the selection of biomarkers can be stated as a *feature selection* problem for a graph classification task.

In this chapter, we adopt a machine learning approach for identifying the most meaningful biomarkers from the structural, functional and multi-modal (MM, i.e., the union of SC and FC), connectomes of healthy controls and

patients. Also, the robustness of the retrieved biomarkers jointly with the classification performance is assessed across different subsamplings of the dataset, allowing us to identify the affected core (a-core) of the pathology.

The results of this chapter appear in biorxiv ¹, as a collaboration with Jakub Vohryzek, Benjamin Chiêm, Philipp S. Baumann, Philippe Conus, Kim Do Cuenod, Patric Hagmann and Jean-Charles Delvenne.

Introduction

Schizophrenia (SZ) is a severe psychiatric disorder characterized by hallucinations and delusions, as well as impairments in memory, attention, executive and other high-order cognitive dysfunctions [182]. The development of magnetic resonance imaging (MRI) has offered an effective way to examine the anatomy of the brain and has motivated numerous scientists to explore the underlying neuropathology of SZ. Over the past few years, advances in high-field structural and functional neuroimaging have made it possible to map the macroscopic neural wiring system of the human brain [84, 173]. This framework allows us to combine a graph theoretical approach with functional and/or diffusion MRI to investigate the brain network alterations occurring in SZ [60, 77, 122]. Studies from the graph theory perspective have shown alterations of both structural and functional brain topology in SZ characterized by a less efficient network organization and the limited capacity of functional integration. Further research using diffusion spectrum imaging reported the brain areas responsible for the loss of network integration and segregation properties [75–77]. However, such findings were identified using conventional univariate strategies performing a separate statistical test at each edge of the connectome under scrutiny, thereby requiring excessively stringent corrections for multiple comparisons. On the other hand, multivariate methods are promising, although they require specialized approaches when the number of parameters dominates the observations [30]. In this study, we adopt a machine learning approach that aims at discovering the most relevant set of biomarkers for discriminating subjects groups and thus quantitatively describing the group differences, both in terms of classification accuracy and stability of selected features.

¹doi: <https://doi.org/10.1101/711135>

Machine learning and biomarker selection

The identification of regions or connections of interest associated with a neural disorder is referred to as biomarker discovery. The identification of such biomarkers in schizophrenia could lead to clinically useful tools for establishing both diagnosis and prognosis. From a machine learning perspective, the choice of biomarkers can be addressed as a feature selection problem, aiming to find a subset of relevant features allowing us to differentiate patients from control subjects accurately.

In this work, we perform an automatic feature selection procedure to identify biomarkers that are relevant for the diagnosis of schizophrenia from brain connectivity data. In this context, biomarkers, therefore, correspond to *structural or functional links* between neural Regions of Interest (ROIs).

A key challenge in feature selection lies in the fact that diverse feature selection methods might result in different sets of retrieved features. Even when using the same technique, it may produce different results when applied to different splittings of the data. When the dimensionality of the input data is high and exceeds the number of training examples, the complexity grows by several orders of magnitude [23, 81]. As a consequence, when two subsets of features are compared the decision of which should be preferred involves uncertainty [88].

These issues underline the need to integrate the stability in the feature selection process so that the method can retrieve consistent features across random subsamplings of the dataset. This is especially true in a biomedical context where many authors have focused on improving the classification performance in several mental disorders such as schizophrenia [121], Alzheimer [47], depression [36]. Even though the stability of biomarker selection has been studied mainly in genomics and proteomics [1, 88], the stability of feature selection has been overlooked in the connectomics community. Therefore, we propose a general framework for stability analysis of selected features, thereby enabling the robust identification of impaired connections in the connectome of schizophrenic patients. The proposed approach is extendable to other brain disorders as well.

In the present work, we use Support Vector Machines (SVM) as a classifier [23] (see Section 2.2.2). This is a supervised machine learning method that aims to classify data points by maximizing the margin between classes in a high-dimensional space. This classifier offers state of the art classification performances on a wide range of applications and is particularly appropriate for

high-dimensional problems with few examples. The SVM classifier has been integrated into an embedded feature selection approach [155]. The so-called Recursive Feature Elimination with Support Vector Machine (RFE-SVM) technique, see Section 2.2.2, was first introduced to perform gene selection for cancer diagnosis on microarray data [83]. More recently, it has also been used on human brain networks to identify differences in structural connectivity related to gender [10]. This method trains an SVM classifier removing the less important features and iteratively re-estimating the classifier with the remaining features until reaching the desired number of them. Accordingly, we adopt the RFE-SVM approach to automatically select brain connections that lead to the best discrimination between patients and controls, and consequently to highlight brain regions that are responsible for the disease.

The aim of the present work is threefold: First, we investigate the effect of structural, functional, and multi-modal (structural+functional) connectome with different resolutions in the classification performance of schizophrenia. Second, we perform a careful feature selection procedure across modalities in order to assess the robustness of the selected features providing the best trade-off between high accuracy and stability. Finally, the analysis of retrieved biomarkers allows us to identify a distributed set of brain regions engaged in the discrimination of patients and control subjects.

This chapter is organized as follows: Section 6.2 introduces the general properties of the dataset as well as the protocol we used in the biomarker identification. In Section 6.3, we present the results on stability, classification performances, and identification of brain areas indicative of the pathology. Finally, in Section 6.4, we lead a discussion on our findings and conclusions.

Dataset and methods

Dataset

For this study, we consider two balanced groups of subjects. The cohort consisted of a schizophrenic group of 37 subjects with a mean age of 40.9 ± 9.4 years and a control group of 37 healthy subjects with a mean age of 32.3 ± 7.6 years. Structural and diffusion MRI data were preprocessed in order to build structural connectomes, i.e., connectivity matrices, for each subject. The structural connectivity between ROIs, i.e., edges of the brain network, were quantified in

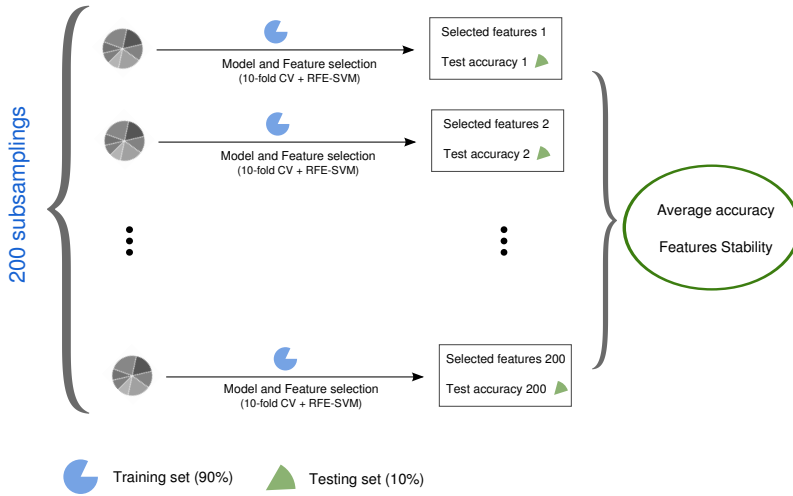


Figure 6.2: Overview of the proposed method. Feature selection is performed systematically across different partitions of the original dataset. Robustness of selected biomarkers is assessed from the output of the RFE-SVM algorithm, and the final accuracy is averaged over subsamplings estimations.

terms of the normalized connection density. Besides, functional MRI BOLD² signals were also preprocessed in order to build functional connectomes for all subjects. Here, a functional connection between ROIs, i.e., an edge, was determined through the Pearson correlation between pairs of BOLD signals. In both cases, the connectivity matrices come in three resolutions: 83×83 , 129×129 and 234×234 as the subcortical regions are included. Thus, for each subject, we have six types of data representing the brain network. For further details about the data collection and preprocessing, we refer the reader to the appendix A.

Biomarker evaluation protocol

Our evaluation methodology is based on Abeel et al. [1] used for biomarker identification in cancer diagnosis on microarray data. In order to assess the robustness of the biomarker selection process, we generate slight variations of the dataset and compare the outcome of selected features across these variations. Therefore, for a stable marker selection algorithm, small variations in the training set should not produce important changes in the retrieved set of features. Concretely, we generate 200 datasets from the original one, repeating 20 times 10-fold cross-validation (CV). Subsequently, for each CV partition,

²Blood-oxygen-level-dependent imaging

90% of folds are used as a training set for selecting the model and features, and the remaining 10% as the testing set to provide an unbiased evaluation of a final model and assess the performance of the classifier. Therefore, the overall accuracy is given by the average testing accuracy across subsamplings. See Figure 6.2 for a schematic view of the methodology.

Assessing the stability of feature selection

We consider the vectorized connectivity matrices of the connectomes as input features for the biomarker selection process. Therefore, for a given connectome, one structural feature refers to the normalized connection density between two linked brain regions, whereas a functional feature refers to the Pearson correlation between two individual brain time courses.

We consider a dataset with $M = 74$ subjects and N features. If we denote the considered connectome resolution as $d \in \{83, 129, 234\}$, the number of features is $N = \frac{d(d+1)}{2}$ because of the symmetry of the connectivity matrices. Drawing $k = 200$ subsamplings and after applying a feature selection procedure (RFE-SVM) in the 90% of each subsampling, we obtain a respective feature signature, i.e. sequence of indices of selected features. Considering two signatures f_i and f_j obtained from different subsamplings i and j , the stability index [105] between f_i and f_j is defined as:

$$KI(f_i, f_j) = \frac{rN - s^2}{s(N - s)} = \frac{r - (s^2/N)}{s - (s^2/N)} \quad (6.1)$$

where $r = |f_i \cap f_j|$ and $s = |f_i| = |f_j|$, the size of the signature, is the number of selected features. This quantity measures the consistency between pairs of features. For s and N fixed KI increases when increases r , reaching its maximum at 1 when the two subsets are identical. The minimum value is bounded from below by -1 when the subsets are perfectly disjoint and the signature size of $N/2$. The overall stability index for a sequence of signatures is defined as the average of all pairwise stability indices on k subsamplings:

$$I_{tot} = \frac{2}{k(k-1)} \sum_{i=1}^{k-1} \sum_{j=i+1}^k KI(f_i, f_j) \quad (6.2)$$

Given that I_{tot} is bounded between -1 and 1 , the greater this value the better the agreement between the selected subsets of features. In particular, a negative value for I_{tot} indicates that the potential agreement between the selected biomarkers is mostly due to chance. In the sequel we will refer to the overall stability I_{tot} as the Kuncheva index [105].

Measuring the classification performance

Because our dataset is class balanced and the task is a binary classification problem, we adopt the accuracy as the metric to quantify the performance of classification. This metric is defined as the ratio of correct classifications to the number of classifications done as follows:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \quad (6.3)$$

where TP is true positive (number of control subjects classified correctly), TN is the true negative (patients classified correctly), FP is false positives (number of patients classified as control subjects) and FN is false negatives (number of control subjects misclassified).

Embedded feature selection (RFE-SVM)

In this work we use recursive feature elimination and support vector machine (RFE-SVM), as feature selector [23, 83]. However, we stress that the methodology can also be applied with other feature selector techniques.

As we have seen in Chapter 2, the RFE-SVM algorithm has a set of internal parameters influencing the computational complexity and the accuracy of the method. The fraction E of features to remove at each step of RFE (also called step size) is critical for the running time. Dropping one feature at a time allows a finer selection but with a prohibitive computational cost. Following Abeel et al. [1], we drop 20% of the least relevant features at each iteration by default. Yet, a stopping criterion is needed to stop the iterative process. Thus, in our experiments, we dropped features until reaching a minimum of $s \in \{0.5, 1, 2, 5, 10, 25, 50\}$ percentage of selected features (stopping criterion).

Another critical parameter is the regularization constant C of the SVM. The C parameter controls the misclassification rate of the classifier, see Eq. 2.29. A larger value makes the optimization choose a smaller margin hyperplane, losing generalization capabilities. The lower the C , the larger the margin of separation, yielding more misclassified points. Therefore this parameter influences the classification accuracy of the model. We cross-validate the optimal C using only the training set, e.g., the 90% of each subsampling.

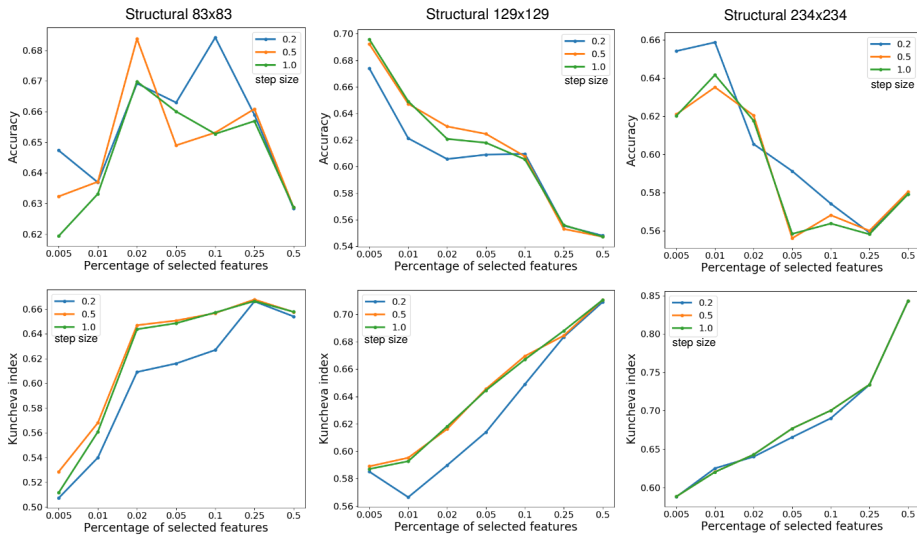


Figure 6.3: Each column represents the average classification accuracy (top) and stability (bottom) for a given scale of the structural connectome. The colors correspond to the percentage of features dropped (step size) at each step of the RFE-SVM algorithm.

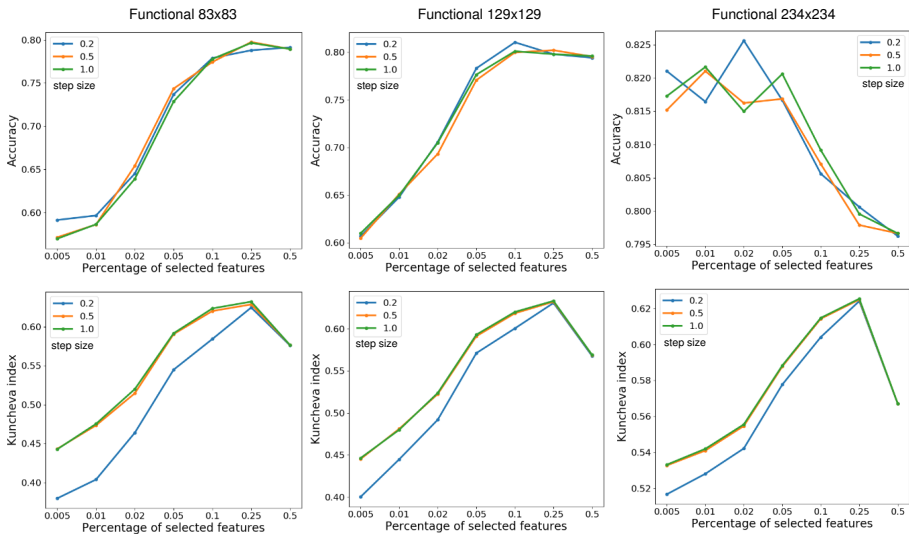


Figure 6.4: Each column represents the average classification accuracy (top) and stability (bottom) for a given scale of the functional connectome. The colors correspond to the percentage of features dropped (step size) at each step of the RFE-SVM algorithm.

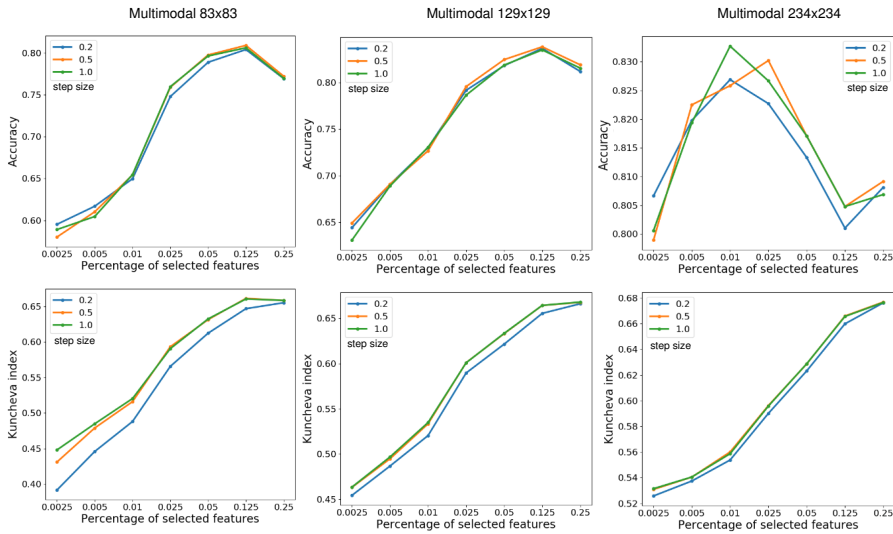


Figure 6.5: Each column represents the average classification accuracy (top) and stability (bottom) for a given scale of the multimodal connectome. The colors correspond to the percentage of features dropped (step size) at each step of the RFE-SVM algorithm.

Results

Connectomes classification and features stability

First, we investigated the effect of different brain connectivity modalities and different scales in the discrimination of patients and normal controls. For each case, we control the step size and the percentage of selected features of the RFE-SVM algorithm, assessing their impact on the classification accuracy and the stability of the selected features.

Figures 6.3, 6.4 and 6.5 show the average classification accuracy after performing RFE-SVM as well as the stability of selected biomarkers across modalities and scales. It can be seen that across scales, the functional connectivity matrices Figure 6.4 achieve better accuracies than the structural matrices Figure 6.3, but conversely, structural matrices are more stable than the functional ones. However, when combining the two modalities, i.e., by concatenating features of both modalities and letting the algorithm choose a blend of structural and functional features, we achieve the best performances Figure 6.5 in terms of both accuracy and stability. Note that the multi-modal curves overlap more than the structural and functional ones in the same resolutions. This shows

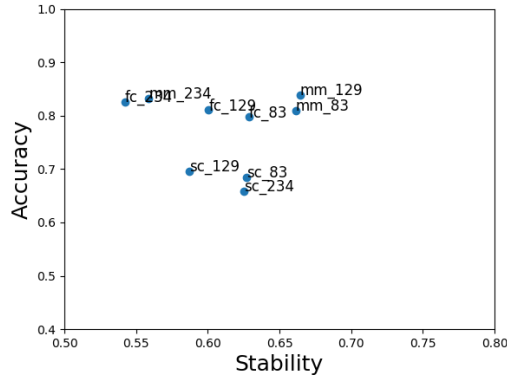


Figure 6.6: Accuracy versus stability for all considered modalities and resolutions.

that the multi-modal matrices provide similar performance in both accuracy and stability concerning the number of dropped features in the RFE-SVM algorithm. It is to be noted that in the multi-modal case, the percentage of finally selected features is divided by two since we combine twice as many features as in the case of structural or functional connectivity alone. In doing so, the stopping criterion will be the same for all modalities. We observe in all cases that the stability increases with the percentage of selected features, which is expected since the overlapping between signatures in Eq. 6.2 is more likely when more features are considered.

To further investigate the effect of modalities and resolutions on both classification accuracy and stability, we extract the best scores from Figures 6.3, 6.4 and 6.5, and plot them in Figure 6.6.

As demonstrated, the best trade-off between both metrics is achieved by multi-modal 129×129 , with accuracy of 0.84 and stability index of 0.67, as well as the multi-modal 83×83 with accuracy of 0.83 and stability index of 0.66. The 234×234 multi-modal resolution has the lowest stability, which can be explained by the fact that finer resolutions are subsamplings of smaller ROIs introducing redundant and correlated edges in the connectome.

Identification of brain regions in schizophrenia

We proceed with the identification of brain areas involved in the classification of patients and controls. For simplicity in the identification of brain regions and comparison with other authors, we analyze the results for the multi-modal

83×83 connectome.

Selected features in the graph space correspond to links representing either connection densities in the structural matrices or Pearson correlations in the functional connectome. Furthermore, inspecting the frequency of each selected feature across subsamplings informs us about the overall relevance of the edge in the classification. In other words, the frequency of an edge is indicative of the importance of the associated ROIs in the classification task.

Given a brain connectivity matrix at a resolution r we define $\mathbf{W}$ as the $r \times r$ matrix where the element $w_{i,j}$ encodes the frequency at which the edge (i, j) is selected as relevant across subsamplings. Thus, the degree of relevance of an ROI i reads:

$$d_i = \sum_{j \in N_i} w_{i,j} \quad (6.4)$$

Figure 6.7 and 6.8 represents the degree of relevance of brain regions for the multi-modal 83×83 connectome sorted in decreasing order. We defined the affected core (a-core) to be composed of brain areas with a degree of relevance higher than the overall average. These findings overlap with the brain regions in the definition of the affected core of Griffa et al. 2015 [75]. For further information see Figure 6.9.

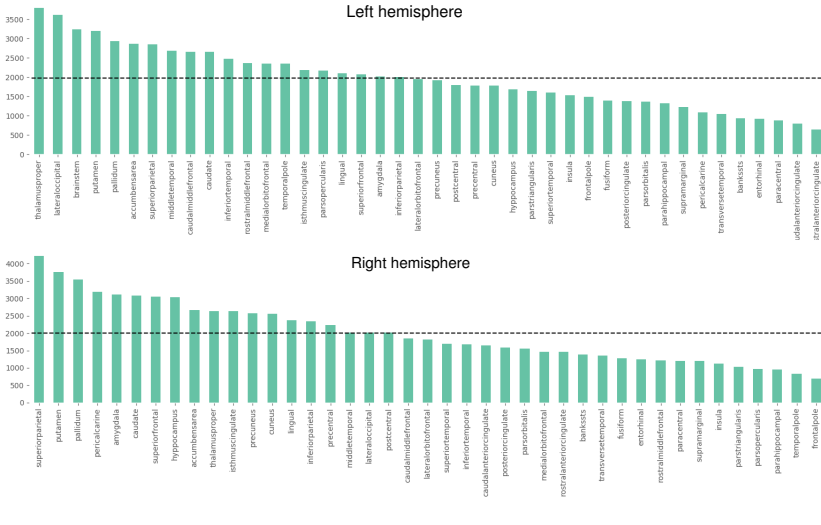


Figure 6.7: Degree of relevance for ROIs in the structural mode of the multimodal 83×83 connectome. The horizontal line is the average degree of relevance.

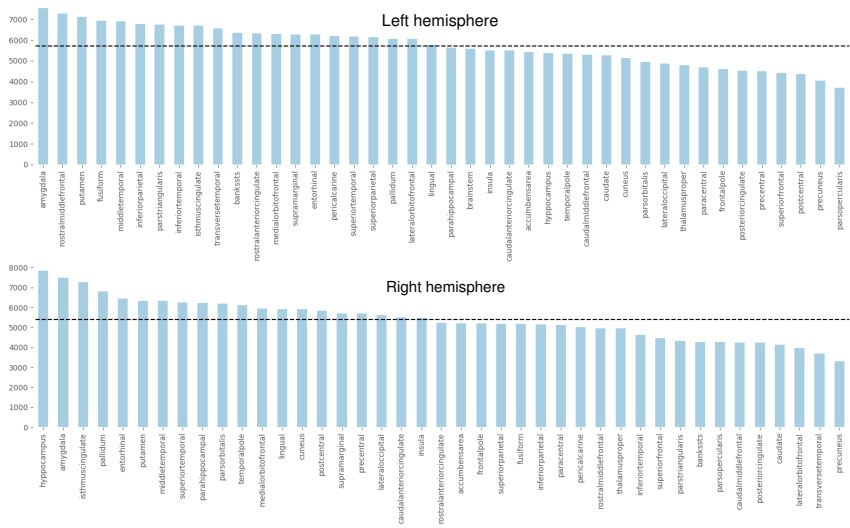


Figure 6.8: Degree of relevance for ROIs in the functional mode of the multimodal 83x83 connectome. The horizontal line is the average degree of relevance.

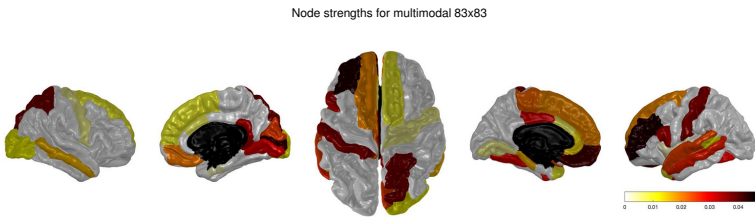


Figure 6.9: Brain surface representation of brain areas with higher relevance degree than the average for the 83x83 resolution of the multi-modal structural and functional mode.

Characterizing intergroup differences

We also investigate the influence of the selected biomarkers in the brain connectivity networks for patients and healthy subjects. To do so, the connectivity strength inside and outside the identified a-core was estimated and compared between both groups. The connectivity strength between ROIs was quantified with two metrics: the inverse apparent diffusion coefficient (iADC) and the generalized fractional anisotropy (gFA). These measures were weighted by the tract size expressed as the number of fibers. Subsequently, they were averaged over all connectivity strengths between a-core nodes and the overall connectivity strengths between regions outside the a-core. We reported the

p -values from the Mann–Whitney–Wilcoxon (MWW) test (see appendix B.2) for between-group differences of the averaged connectivity strengths, with a significance level $\alpha = 0.05$.

As can be seen in Figure 6.10, the weighted gFA and weighted iADC were altered in patients compared with controls. When averaging within the a-core the gFA was decreased with p -value = 0.00017588 as well the iADC with p -value = 0.018435. None difference between groups was found in the outside a-core network of averaged gFA and iADC.

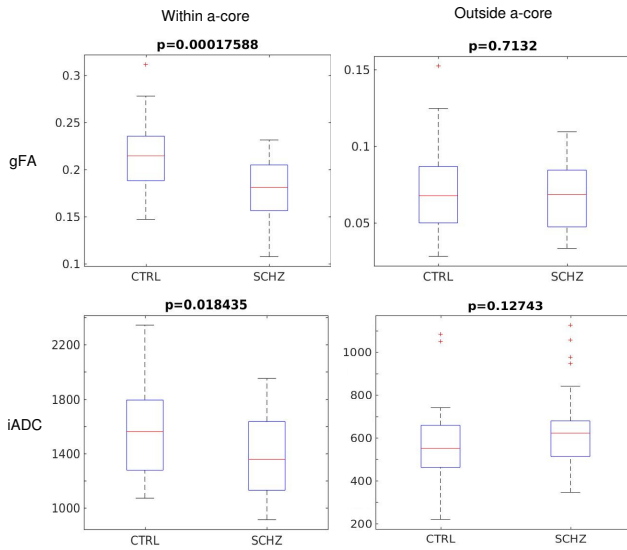


Figure 6.10: Boxplots of the distribution of connectivity strength averaged within (left) and outside the a-core. Both considered metrics, the generalized fractional anisotropy (gFA) and inverse apparent diffusion coefficient (ADC) were weighed by the size of the tracts.

Conclusions and perspectives

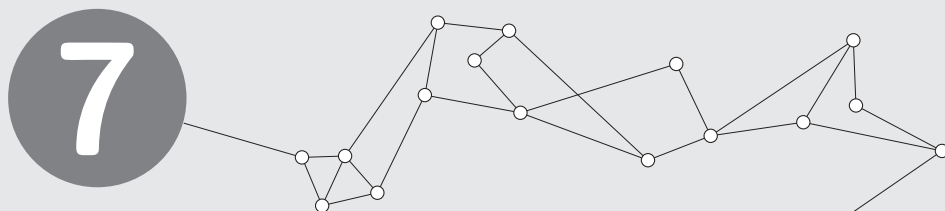
This chapter has investigated the effect of different connectivity modes of the human connectome in the identification of robust biomarkers for the diagnosis of schizophrenia. We perform an automatic feature selection process on the edge space aiming to retrieve a compact subset of meaningful biomarkers performing accurately on the diagnosis of schizophrenia. In addition, we analyze the robustness of the retrieved features concerning the sample variation, based on the fact that stable biomarkers will not change dramatically in dif-

ferent subsamplings of the underlying dataset [1]. It turns out that combining structural and functional connectivity matrices as a multi-modal representation of connectomes provides the best trade-off between high accurate and stable biomarkers.

Importantly, the fact that the best accuracy and stability was achieved by combining SC and FC features reflects the relevance of both approaches to studying and classifying brain disorders. It contributes to the literature on the relationship between structural and functional large-scale networks [78,85] and highlights the importance of understanding both approaches to explain complex neural disorders at least insofar the prediction of them is concerned.

Based on the frequency from which the RFE-SVM algorithm selects relevant edges, we can map edges, i.e., brain connections, to the node space by looking at the strengths of connection densities or correlation between brain regions. The degree of relevance for nodes allows us to identify the affected core as the brain regions that are highly active during the training phase and therefore, are selected more often for relevant edges. Our findings overlap results from Griffa et al. 2015 [75] and provide further evidence of brain regions involved in the pathology. Furthermore, they offer a unique perspective in defining the affected core both from the influence of SC and FC and hence establishing an SC-FC a-core which might give a more detailed description of the regions that are affected in SCHZ [75,122].

The alterations of gFA and iADC values within and outside the a-core depicted in Figure 6.10 suggest a micro-structural impairment of the a-core regions given by an interruption of white matter across the affected core circuits.



Conclusions and perspectives

Summary and conclusions

This thesis has discussed a special range of topics relating to machine learning and complex networks. Across the chapters, we have addressed essential concerns in graph representation for machine learning applications and feature engineering, using dynamics on networks as a proxy to perform ML on graph-based data. We have developed new complex network tools and implemented diverse applications on graph classification, clustering, anomaly detection on networks, graph visualization, and graph representation learning.

Although the chapters already included a discussion about their specific topics, here we will briefly recap some results pointing out some issues with opportunities of further work.

In Chapter 3, we presented an innovative multi-scale anomaly detection algorithm for attributed networks. As far as we know, this is the first proposed algorithm capable of uncovering anomalous nodes in a multi-scale fashion, i.e., anomalous nodes in different levels of hierarchical clusters (contexts) on the network. The key feature of the method relies on the link between graph signal processing and heat diffusion on networks. While filtering signals on graphs allows spotting anomalous nodes, diffusion of heat across the network uncovers the context of anomalousness for a given node using the Markov stability framework [51].

Chapter 4 introduces a generalized definition of assortativity mixing on networks. Understanding higher-order correlations between nodes attributes may reveal insights into the strength of social ties in multiple scales. Random walk dynamics provides a natural description of such multi-hop correlations between node meta-data. We showed that using multi-hop assortativities as graph features provides superior performance in graph classification, beating representative graph-kernels, and some neural network approaches. Besides, we believe that assortativity can be extended in a straightforward manner in-

roducing correlations between multidimensional node attributes, e.g., vectors of node attributes, remaining as an interesting topic for further research.

We also provide an unsupervised way to uncover the hidden structure over a collection of networks defined on the same set of nodes. We have shown in Chapter 5 that learning graph embeddings from networks allows capturing the intrinsic similarity between graphs. This graph representation is useful for ML learning tasks such as graph classification, clustering, and visualization. However, the autoencoder approach we employed has the restriction of requiring networks defined on the same set of nodes. Furthermore, additional research towards graph representation learning in general graphs needs to be conducted. In addition, recent deep learning developments operating on graphs such as Graph Convolutional Neural Networks (GCNN) [49,101] and Variational Graph Autoencoders (VGAE) [102,118] are trending nowadays. Building on top of them to design new graph representation learning approaches leveraging node meta-data, offers an exciting direction for further research.

Finally, in Chapter 6, we addressed the problem of evaluating the stability of feature selection for the identification of biomarkers in schizophrenia. Although the stability of feature selection has been studied broadly in other fields such as genomics [1,88], it has been overlooked by the connectomics community. Our research shows that the mix of structural and functional modes, i.e., multi-modal brain connectivity matrices, offers the best trade-off between stable biomarkers and higher accuracy in schizophrenia prediction. This highlights the significance of structure and function in the understanding of the neural illness. Moreover, stable biomarkers allowed us to identify the affected core in schizophrenia, as a subset of ROIs with decreasing connectivity strength in patients, confirming previous findings in the literature. Therefore, we hope our results are encouraging and will promote future applications in Alzheimer [47], Parkinson [65], Depression [36] and other neural disorders.

Perspectives

Beyond the contributions presented in this thesis, the spectrum of ML in complex networks has enormous potential for new developments and opportunities for further research. As we have previously mentioned some open issues for future work, here we speculate about that could be the next wave of developments in the field.

Human-aware graph representations

A trending topic in ML research on networks is the development of graph-related embeddings. A vast number of such methods have been proposed during the last decade [32, 71, 79, 134, 149, 176], differing mainly in the embedded structure, i.e., nodes, edges, communities, subgraphs, graphs, paths, cliques, and other patterns. Furthermore, many methods can also incorporate external meta-data, i.e., node or edge attributes, producing excellent network representations used in down-stream ML tasks such as node classification, link prediction, clustering, and graph classification.

However, in many real-world applications, especially those involving social networks, there is sensitive information that should be controlled within the embeddings. For instance, in an online professional network such as LinkedIn, one may desire that job recommendations are fair or balanced concerning specific attributes such as gender, religion, or race. In other words, embeddings are expected to be invariant concerning sensitive meta-data. We stress that, in addition to preserving structural properties, graph-related embeddings should incorporate fairness constraints, ensuring that the representations are not correlated with certain attributes that may bias the outcome of ML tasks.

Besides, the increasing concerns about privacy and data protection will require graph-related embeddings to include mechanisms to prevent unveiling hidden data. For instance, in social media such as Facebook or Twitter, a user may want to mark a part of its ego-network as private, then edge embeddings in their network should be protected from any link prediction effort. Thus, the challenges here will turn around protecting individual privacy without compromising functionality.

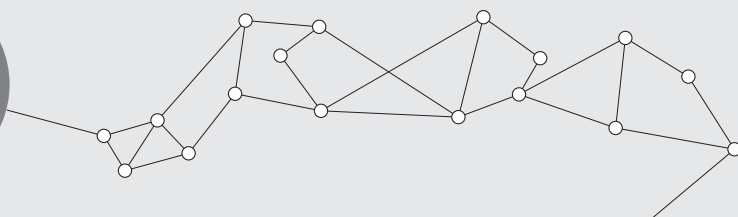
Deep generative models in graphs

With the breakthrough of deep generative adversarial networks (GANs) [70], the popularity of generative models exploded during the last decade. This powerful model allows learning complex data distribution in an unsupervised fashion with astonishing results in image [70], speech [180], art, and text generation [58]. The developing of deep generative models on graphs have gained much attention, lastly, in the research community. Novel approaches, such as graph generative adversarial networks (GraphGANs) [188] or variational graph autoencoders (VGAEs) [102], allow to learn the wiring rules from graph-based data without human intervention, uncovering the mechanisms by which it grows

and develops.

For instance, applications in molecule generation in chemoinformatics and drug discovery [120, 194] are revolutionizing the field by designing high accurate systems capable of discovering new compounds and materials. As in neuroscience, gathering, cleaning, and processing MRI/fMRI data is still a costly and time-consuming process, it may potentially be overcome with appropriated generative models on connectome data. Thus, these approaches offer a high potential for new developments in the field and it is worth considering further research.

Appendices



Brain data preprocessing

In this section you can find the details about the data collection and preprocessing of the brain data we used in our study presented in Section 6 of this thesis.

Subjects

For this study, two age-balanced groups were considered. The cohort consisted of a schizophrenic group of 37 subjects with a mean age of 40.9 ± 9.4 years and a control group of 37 healthy subjects with a mean age of 32.3 ± 7.6 years. The patients in the schizophrenic group were recruited from the psychiatric clinic at the Lausanne University Hospital. They met DSM-IV criteria for schizophrenic and schizoaffective disorders (American psychiatry association, 2000 ref). We obtained a written consent from all the subjects following the institutional guidelines approved by the Ethics Committee of Clinical Research of the Faculty of Biology and Medicine, University of Lausanne, Switzerland.

Brain network estimation

A general scheme in the extraction of brain networks from empirical data is shown in Figure A.1. The details about the inter-media steps are detailed in the following.

Magnetic resonance imaging

All subjects were scanned on the 3 Tesla Siemens Trio scanner with a 32-channel head coil. Three acquisition protocols were part of the MRI session: 1) magnetization-prepared rapid acquisition gradient echo (MPRAGE) sequence with in-plane resolution of 1 mm, slice thickness of 1.2 mm of total voxel

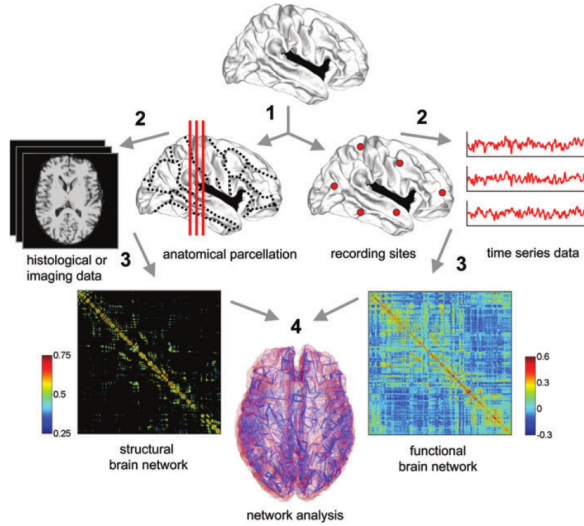


Figure A.1: Extraction of brain networks from empirical data follows along four steps.(1) parcellation of the brain volume into coherent regions on the basis of structural or connectional features (MRI), or node assignment by placement of sensors and/or recording sites (EEG, BOLD signals); (2) structural or diffusion imaging to derive estimates of structural connectivity (left) or recording of time series data to estimate functional coupling (right); (3) construction of a connection matrix representing a structural (left) or functional network (right); and (4) network analysis. Figure taken from [172]

number of $240 \times 257 \times 160$ and TR, TE and TI were 2300, 2.98 and 900 ms respectively, 2) diffusion spectrum imaging (DSI) sequence with 128 diffusion-weighted images of b0 as a reference image and a maximum b-value of 8000 s/mm². The time of acquisition was 13 min and 27s. The number of voxels was $96 \times 96 \times 34$ with a resolution of $2.2 \times 2.2 \times 3.3$ mm, and TR and TE were 6100 and 144 ms respectively. The issue of motion- artifacts linked to signal drop-outs was dealt with by visually inspecting the signal, and no subject had to be excluded as a result of this [196].

Structural networks

Structural and diffusion MRI data were used to estimate the weighted and undirected structural connectivity matrices in the Connectome Mapping Tool-kit [33, 46, 75]. Firstly, white matter, grey matter, and cerebrospinal fluid segmentation was performed on the structural data and further linearly registered to the b0 volumes of the DSI dataset. Secondly, the first three scales

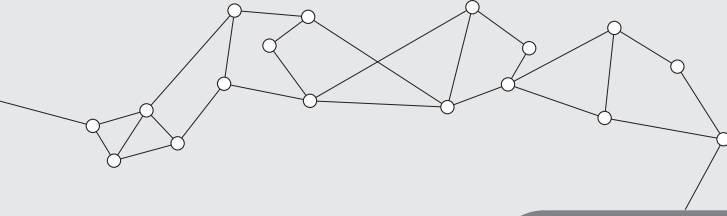
of the Lausanne multi-scale atlas were used to parcellate the grey matter. In detail, the first scale consisted of 68 cortical brain regions and 14 subcortical regions with scale two and three subdividing the first scale into 114 and 219 cortical regions [33]. Further, deterministic streamline tractography, estimating 32 diffusion directions per voxels, was used to reconstruct the structural connectivities from the DSI data [189]. The normalized connection density quantified the structural connectivity between brain regions and is defined as follows,

$$w_{ij} = \frac{2}{S_i S_j} \sum_{f \in E_f} \frac{1}{\ell(f)} \quad (\text{A.1})$$

where $w_{i,j}$ represents an edge between brain regions i and j , S_i and S_j are the surface areas of regions i and j , $f \in E_f$ represents a streamline f in the set of streamlines E_f and $\ell(f)$ is a length of a given streamline f [75, 84]. The normalisation by brain region surfaces accounts for their slightly varying size and the streamline length normalisation accounts for a bias towards longer connections imposed by the tractography algorithm.

Functional networks

Functional connectivity matrices were computed from fMRI BOLD time series. Firstly, the first four-time points were excluded, yielding the number of time points to be $T = 276$ [94]. Rigid-body registration was applied to individual timeslices for motion-correction. The signal was then linearly detrended and corrected for physiological confounds and further motion artefacts by regressing white-matter, cerebrospinal fluid, and six motion (translations and rotations) signals. Lastly, the signal was spatially smoothed and bandpass-filtered between 0.01 – 0.1 Hz with Hamming windowed sinc FIR filter. Linear registration was performed between the average fMRI and MPRAGE images to obtain the ROIs time series [95]. An average timecourse for each brain region was computed for the three atlas scales. We build the functional matrices by computing the absolute value of the Pearson correlation between individual brain regions' timecourses. All of the above was performed in subject native space with Connectome Mapper Toolkit and personalized Python and Matlab scripts [46] [78].



Statistical tests

Friedman-Nemenyi test

The t -test is a widely used statistical method to compare classifiers. Nevertheless, It is limited to work with variables being commensurable and normally distributed. Besides, the test is affected by outlier measurements [52].

The Friedman test and Nemenyi post-hoc test do not exhibit the previous weakness, and can be used to compare multiple classifiers. The Friedman test provides evidence that the outcome of different models is statistically different. Subsequently, the Nemenyi post-hoc test assesses significant differences between individual classifiers (algorithms).

In Chapter 4, we used the Friedman test [64] to compare the accuracies of different algorithms. The Friedman test is a non-parametric test based on the average ranked performances (R_j) of the classification performance on each dataset and is calculated as:

$$Q = \frac{12D}{K(K+1)} \sum_{j=1}^K \left(R_j - \frac{K+1}{2} \right)^2 \quad (\text{B.1})$$

where D denotes the number of datasets, K the number of algorithms and $R_j = \frac{1}{D} \sum_{i=1}^D r_i^j$ as the average rank (AR) of algorithms with r_i^j denoting rank of the j -th of K algorithms on the i -th of N datasets. If all the algorithms perform equally well (null-hypothesis) then we can expect that Q is approximately distributed as a Chi-square distribution with $K - 1$ degrees of freedom. Therefore we can reject the null hypothesis and conclude that some algorithms perform better than other when Q is large, with the probability that $\chi_{K-1}^2 \geq Q$ as p -value.

If the null-hypothesis is rejected, we can proceed with a post-hoc test. The post hoc Nemenyi test [135] is applied to report any significant difference between individual algorithms. The Nemenyi test states that the performance of various algorithms are significantly different if their average rank differ by at

least the critical difference:

$$CD = q_{\alpha,K} \sqrt{\frac{K(K+1)}{12D}} \quad (\text{B.2})$$

where the critical values $q_{\alpha,K}$ are based on the Studentized range statistic divided by $\sqrt{2}$. Finally the results from Friedman-Nemenyi tests are displayed using the diagrams proposed by Demsar [52]. These diagrams show the ranked performances of the classification techniques along with the critical difference to stand out the algorithms which are significantly different to the best performing ones.

Mann-Whitney-Wilcoxon (MWW) test

The Mann-Whitney-Wilcoxon test, used in Chapter 6, sometimes called the Mann-Whitney U test or the Wilcoxon Rank sum test [124], is a nonparametric test for testing whether one of two random variables is stochastically larger than the other. This test can be used to investigate whether two independent samples were selected from populations having the same distribution. For two randomly drawn observations x and y taken from different populations, the research hypothesis states:

$$\begin{aligned} H_0 : \quad & \text{Prob}(x > y) = 0.5 \\ H_1 : \quad & \text{Prob}(x > y) \neq 0.5 \end{aligned} \quad (\text{B.3})$$

This test is performed often as a two-sided test. Given two populations of sizes n_1 and n_2 , the procedure for the test involves joining the observations from both populations into one combined sample, keeping track of which sample each observation comes from, and ranking them from lower to highest from 1 to $n_1 + n_2$ respectively. The test statistic for the MWW test is denoted by U and is defined as $U = \min\{U_1, U_2\}$, where U_1 and U_2 are defined as:

$$\begin{aligned} U_1 &= n_1 n_2 + \frac{n_1(n_1 + 1)}{2} - R_1 \\ U_2 &= n_1 n_2 + \frac{n_2(n_2 + 1)}{2} - R_2 \end{aligned} \quad (\text{B.4})$$

where R_1 and R_2 are the sum of the ranks for the groups 1 and 2 respectively. The critical value is determined in terms of the sample sizes and a desired level of significance, and is taken from a table. We refer the reader to the Ref. [124] for more details about the method.

In the MWW test, the theoretical U varies from 0 (complete separation between groups, H_0 most likely false and H_1 most likely true), to $n_1 \times n_2$, where the evidence to support H_1 is insufficient. The null hypothesis is therefore, rejected if the critical value is larger or equal to the test statistic U .

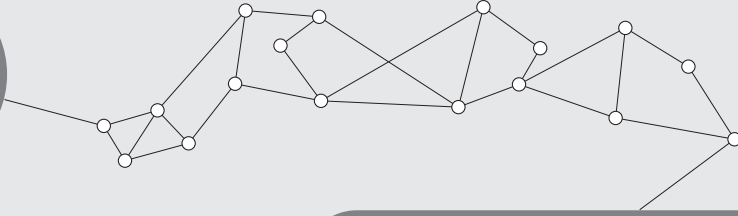
Pearson correlation coefficient

Pearson's correlation coefficient is a statistical measure used to quantify the strength of a linear relationship between paired variables. It is bounded from below by -1 and from above by 1 . Positive values indicate a tendency of one variable to increase or decrease together with another variable. Negative values of correlation coefficient indicate a tendency that the increase of values of one variable is associated with the decrease of values of the other variable and vice versa. Correlation coefficient equal to 1 means perfect positive linear correlation, 0 denotes no linear correlation and -1 a perfect negative linear correlation.

The Pearson correlation coefficient is denoted by r . Thus, for a paired sample $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$ consisting of n observations, the Pearson coefficient is computed as:

$$r = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}} \quad (\text{B.5})$$

where $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$ is the sample mean; and analogous for $\bar{y}$.



Parameters of experiments

Multi-hop assortativities

The parameters for the graph-kernels approaches used in Chapter 4 are cross-validated on the training set following [163] and [195] settings:

- The p value for p -step random walk kernel is chosen from $\{1, 2, 3\}$
- We computed the random walk kernel for the decay $\lambda \in \{10^{-6}, 10^{-5}, \dots, 10^{-1}\}$
- The height parameter in Ramon & Gartner's kernel is taken from $\{1, 2, 3\}$
- In the Deep Graph kernels (GK, SP and WL), the window size and feature dimension is chosen from $\{2, 5, 10, 25, 50\}$
- Similar to other works [195], [142] we set $h = 2$ for Weisfeiler-Lehman subtree kernel

Unsupervised Network Embeddings

For completeness we provide the parameters we used in our experiments of Chapter 5.

Denoising autoencoder parameters

- Let A the adjacency matrix of a graph. We consider A^3 as input for the DAE.
- Denoting as N the number of nodes of the considered graph, the neural network architecture is:
- We inject 10% of random edges to each example in order to introduce corrupted examples for the denoising process.

Layer	Number of neurons	Activation function
Input	$\frac{N(N+1)}{2}$	—
Hidden	800	$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$
Output	$\frac{N(N+1)}{2}$	sigmoid: $\sigma(x) = \frac{1}{1+e^{-x}}$

- Optimization: Adam algorithm with initial learning rate of 0.001, a batch of 128 examples and 15 epochs.

Graph2vec parameters

We used the following parameters: Batch size: 128, epochs: 15, embedding size: 800, Number of negative samples to be used in training: 10, learning rate: 0.3, degree of rooted subgraph (for WL kernel): 3.

GAM parameters

As this method requires explicitly node attributes, we used the node degree as structural attribute. Optimization was done with the following parameters: Number of training epochs 15, number of neurons in the step network as 32, number of neurons in the shared layer of the step net 64, batch size of 128, time budget for steps 10, learning rate of 0.001 and weight decay 10^{-5} .

Subgraph2vec parameters

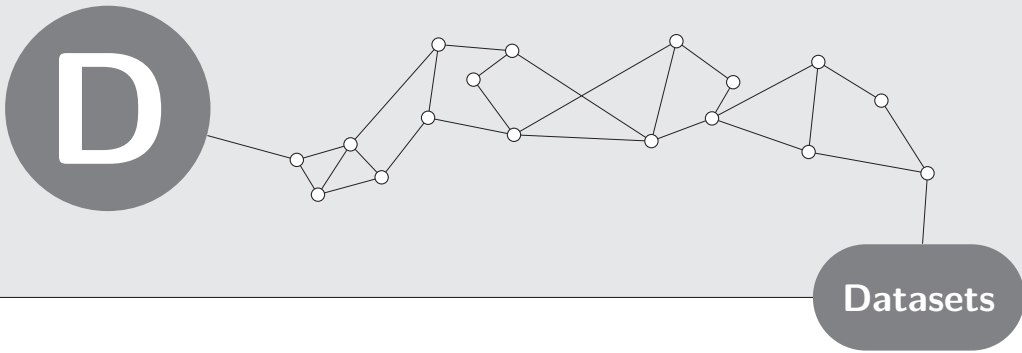
The parameter used for training this model were: batch size of 128, epochs 3, embedding size 800, learning rate for optimization 1.0, negative sub sampling of 10 and weight of WL kernel 3.

Synthetic datasets

In the following we provide the parameters used to generate the synthetic datasets. The considered number of nodes is $N = 81$ and 600 examples for each case.

- ER: Erdős-Rényi networks. We generate networks with average degree equal to 4, 6, 8 and 10. Parameter $p \in \{0.05, 0.075, 0.10, 0.125\}$
- Mixed: Power law and random networks. We use the Barabási-Albert model with $m = 3$ and Erdős-Rényi model with $p = 0.075$, generating networks with an average degree of six.

- LFR: we generate power law networks with $\mu = 0.1$ and $\mu = 0.5$. The average node degree is 11, community sizes between 6 and 22 nodes, the exponent for the degree distribution equal to 2 and the exponent for community size distribution of 1.
- Dynamic: initial random network with $p = 0.08$. We remove and add edges with a probability of 0.015. At time $t = 200$ we remove and add edges with probability of 0.2, increasing it at time $t = 400$ to probability of 0.6



Here we list the real life datasets we used in our thesis. All urls where checked on July 2019.

Anomaly detection on attributed networks

- **Disney**
<https://www.ipd.kit.edu/~muellere/consub/RealData/Disney.zip>
- **Books**
<https://www.ipd.kit.edu/~muellere/consub/RealData/AmazonFail.zip>
- **Enron**
<https://www.ipd.kit.edu/~muellere/consub/RealData/Enron.zip>

Multi-hop assortativities in networks

- **MUTAG, PTC, NCI1, NCI109, ENZYMES, PROTEINS, IMDB_BINARY, IMDB_MULTI**
<https://github.com/leoguti85/MaF/blob/master/datasets/>
- **COLLAB, REDDIT_BINARY, REDDIT_MULTI5K, REDDIT_MULTI10K**
<https://bit.ly/2Ygwpsx>

Network embeddings with node identity awareness

- **Primary school temporal network**
<http://www.sociopatterns.org/datasets/primary-school-temporal-network-data/>
- **Air transportation multiplex network**
<http://complex.unizar.es/atnmultiplex/>

- [1] T. Abeel, T. Helleputte, Y. Van de Peer, P. Dupont, and Y. Saeys. Robust biomarker identification for cancer diagnosis with ensemble feature selection methods. *Bioinformatics*, 26(3):392–398, 11 2009.
- [2] C. C. Aggarwal. *Outlier Analysis*. Springer Publishing Company, Incorporated, 2013.
- [3] C. C. Aggarwal. Outlier ensembles: Position paper. *SIGKDD Explor. Newsl.*, 14(2):49–58, Apr. 2013.
- [4] L. Akoglu, M. McGlohon, and C. Faloutsos. oddball: Spotting anomalies in weighted graphs. In M. J. Zaki, J. X. Yu, B. Ravindran, and V. Pudi, editors, *Advances in Knowledge Discovery and Data Mining*, pages 410–421, 2010.
- [5] L. Akoglu, H. Tong, and D. Koutra. Graph based anomaly detection and description: A survey. *Data Min. Knowl. Discov.*, 29(3):626–688, 2015.
- [6] R. Albert and A.-L. Barabási. Statistical mechanics of complex networks. *Rev. Mod. Phys.*, 74:47–97, Jan 2002.
- [7] R. Aldecoa and I. Marín. Deciphering network community structure by surprise. *PLOS ONE*, 6(9):1–8, 09 2011.
- [8] D. Aldous and J. A. Fill. Reversible markov chains and random walks on graphs, 2002. Unfinished monograph, recompiled 2014, available at <http://www.stat.berkeley.edu/~aldous/RWG/book.html>.
- [9] N. Alon and V. Milman. λ 1, isoperimetric inequalities for graphs, and superconcentrators. *Journal of Combinatorial Theory, Series B*, 38(1):73–88, 1985.

- [10] J. C. D. B. Chiêm, F. Crevecoeur. Supervised classification of structural brain networks reveals gender differences. In *2018 19th IEEE Mediterranean Electrotechnical Conference (MELECON)*, 2018.
- [11] A. C. Bahnsen, D. Aouada, A. Stojanovic, and B. Ottersten. Feature engineering strategies for credit card fraud detection. *Expert Systems with Applications*, 51:134 – 142, 2016.
- [12] A.-L. Barabási and M. Pósfai. *Network science*. 2016.
- [13] I. Barnett, N. Malik, M. L. Kuijjer, P. J. Mucha, and J.-P. Onnela. Feature-based classification of networks. Oct. 2016.
- [14] A. Ben-Hur, C. S. Ong, S. Sonnenburg, B. Schölkopf, and G. Rätsch. Support vector machines and kernels for computational biology. *PLOS Computational Biology*, 4(10):1–10, 10 2008.
- [15] C. Berge. *The theory of graphs and its applications*. Methuen ; Wiley, London; New York, 1962.
- [16] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.
- [17] V. D. Blondel, A. Decuyper, and G. Krings. A survey of results on mobile phone datasets analysis. *EPJ Data Science*, 4(1):10, Aug 2015.
- [18] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, oct 2008.
- [19] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D. U. Hwang. Complex networks: Structure and dynamics. *Physics Reports*, 424(4-5):175–308, feb 2006.
- [20] P. Bonacich and P. Lloyd. Eigenvector-like measures of centrality for asymmetric relations. *Social Networks*, 23(3):191–201, July 2001.
- [21] K. Borgwardt and H. Kriegel. Shortest-path kernels on graphs. *Fifth IEEE International Conference on Data Mining (ICDM-05)*.
- [22] K. M. Borgwardt, C. S. Ong, S. Schonauer, S. V. N. Vishwanathan, A. J. Smola, and H.-P. Kriegel. Protein function prediction via graph kernels. *Bioinformatics*, 21(Suppl 1):i47–i56, Jan 2005.

- [23] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pages 144–152, New York, NY, USA, 1992. ACM.
- [24] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner. On modularity clustering. *IEEE Transactions on Knowledge and Data Engineering*, 20(2):172–188, Feb 2008.
- [25] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001.
- [26] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. Lof: Identifying density-based local outliers. *SIGMOD Rec.*, 29(2):93–104, May 2000.
- [27] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, Apr 1998.
- [28] P. Brémaud. *Markov Chains*. Springer New York, 1999.
- [29] M. Budnik, E. Gutierrez-Gomez, B. Safadi, and G. Quénot. Learned features versus engineered features for semantic video indexing. In *2015 13th International Workshop on Content-Based Multimedia Indexing (CBMI)*, pages 1–6, June 2015.
- [30] P. Buhlmann and S. van de Geer. *Statistics for High-Dimensional Data: Methods, Theory and Applications*. Springer Publishing Company, Incorporated, 1st edition, 2011.
- [31] N. D. Cahill. Normalized measures of mutual information with general definitions of entropy for multimodal image registration. In B. Fischer, B. M. Dawant, and C. Lorenz, editors, *Biomedical Image Registration*, pages 258–268, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.
- [32] H. Cai, V. W. Zheng, and K. C. Chang. A comprehensive survey of graph embedding: Problems, techniques and applications. *CoRR*, abs/1709.07604, 2017.
- [33] L. Cammoun, X. Gigandet, D. Meskaldji, J. P. Thiran, O. Sporns, K. Q. Do, P. Maeder, R. Meuli, and P. Hagmann. Mapping the human connectome at multiple scales with diffusion spectrum MRI. *Journal of Neuroscience Methods*, 203(2):386–397, 2012.

- [34] A. Cardillo, J. Gómez-Gardeñes, M. Zanin, M. Romance, D. Papo, F. del Pozo, and S. Boccaletti. Emergence of network features from multiplexity. *Scientific Reports*, 3, 2013.
- [35] C. Castillo, D. Donato, A. Gionis, V. Murdock, and F. Silvestri. Know your neighbors: Web spam detection using the web topology. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, pages 423–430, New York, NY, USA, 2007. ACM.
- [36] M. Chi, S. Guo, Y. Ning, J. Li, H. Qi, M. Gao, X. Wu, J. Xue, X. Du, J. Wang, X. Hu, Y. Guo, Y. Yang, H. Peng, and K. Wu. Discriminative analysis of major depressive disorder and anxious depression using support vector machine. *Journal of Computational and Theoretical Nanoscience*, 12(7):1395–1401, 2015.
- [37] H. Choi, K. Cho, and Y. Bengio. Fine-grained attention mechanism for neural machine translation. *Neurocomputing*, 284:171–176, 2018.
- [38] Y. Chun and D. A. Griffith. *Spatial Statistics and Geostatistics: Theory and Applications for Geographic Information Science and Technology (SAGE Advances in Geographic Information Science and Technology Series)*. SAGE Publications Ltd, 2013.
- [39] F. Chung. Laplacians and the cheeger inequality for directed graphs. *Annals of Combinatorics*, 9(1):1–19, Apr 2005.
- [40] F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.
- [41] T. J. Cleophas, A. H. Zwinderman, and H. I. Cleophas-Allers. *Machine learning in medicine*. Springer, 2013.
- [42] J. Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46, 1960.
- [43] R. R. Coifman and S. Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):5 – 30, 2006. Special Issue: Diffusion Maps and Wavelets.
- [44] C. Cortes and V. Vapnik. Support-vector networks. *Mach. Learn.*, 20(3):273–297, Sept. 1995.

- [45] F. Costa and K. De Grave. Fast neighborhood subgraph pairwise distance kernel. In *Proceedings of the 26th International Conference on Machine Learning, International Conference on Machine Learning, Haifa, Israel, 21-24 June 2010*, pages 255–262. Omnipress, 2010.
- [46] A. Daducci, S. Gerhard, A. Griffa, A. Lemkaddem, L. Cammoun, X. Gigandet, R. Meuli, P. Hagmann, and J. P. Thiran. The Connectome Mapper: An Open-Source Processing Pipeline to Map Connectomes with MRI. *PLoS ONE*, 7(12):e48121, 12 2012.
- [47] Z. Dai, C. Yan, Z. Wang, J. Wang, M. Xia, K. Li, and Y. He. Discriminative analysis of early alzheimer’s disease using multi-modal imaging and multi-level characterization with multi-classifier (m3). *NeuroImage*, 59(3):2187 – 2195, 2012.
- [48] A. K. Debnath, R. L. Lopez de Compadre, G. Debnath, A. J. Shusterman, and C. Hansch. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. *Journal of Medicinal Chemistry*, 34(2):786–797, 1991.
- [49] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 3844–3852. Curran Associates, Inc., 2016.
- [50] J.-C. Delvenne, M. T. Schaub, S. N. Yaliraki, and M. Barahona. *The Stability of a Graph Partition: A Dynamics-Based Framework for Community Detection*, pages 221–242. Springer New York, New York, NY, 2013.
- [51] J.-C. Delvenne, S. N. Yaliraki, and M. Barahona. Stability of graph communities across time scales. *Proceedings of the National Academy of Sciences*, 107(29):12755–12760, 2010.
- [52] J. Demsar. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, Dec. 2006.
- [53] Q. Ding, N. Katenka, P. Barford, E. Kolaczyk, and M. Crovella. Intrusion as (anti)social communication: Characterization and detection.

- In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 886–894, New York, NY, USA, 2012. ACM.
- [54] C. Donnat and S. Holmes. Tracking network dynamics: a survey of distances and similarity metrics. *ArXiv e-prints*, Jan. 2018.
 - [55] J. A. Dunne, R. J. Williams, and N. D. Martinez. Food-web structure and network theory: The role of connectance and size. *Proceedings of the National Academy of Sciences*, 99(20):12917–12922, 2002.
 - [56] W. E. T. Li, and E. Vanden-Eijnden. Optimal partition and effective dynamics of complex networks. *Proceedings of the National Academy of Sciences*, 105(23):7907–7912, 2008.
 - [57] L. Euler. Solutio problematis ad geometriam situs pertinentis. *Eneström*, 53, 1741.
 - [58] W. Fedus, I. Goodfellow, and A. Dai. Maskgan: Better text generation via filling in the. 2018.
 - [59] M. Fiedler. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, 23(2):298–305, 1973.
 - [60] A. Fornito, A. Zalesky, C. Pantelis, and E. T. Bullmore. Schizophrenia, neuroimaging and connectomics. *NeuroImage*, 62(4):2296 – 2314, 2012. Connectivity.
 - [61] S. Fortunato and A. Lancichinetti. Community detection algorithms: A comparative analysis: Invited presentation, extended abstract. In *Proceedings of the Fourth International ICST Conference on Performance Evaluation Methodologies and Tools*, VALUETOOLS '09, pages 27:1–27:2, ICST, Brussels, Belgium, Belgium, 2009. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).
 - [62] F. Fouss, A. Pirotte, J. Renders, and M. Saeuens. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 19(3):355–369, March 2007.
 - [63] F. Fouss, M. Saeuens, and M. Shimbo. *Algorithms and Models for Network Data and Link Analysis*. Cambridge University Press, New York, NY, USA, 1st edition, 2016.

- [64] M. Friedman. A comparison of alternative tests of significance for the problem of m rankings. *Ann. Math. Statist.*, 11(1):86–92, 03 1940.
- [65] S. Galantucci, F. Agosta, E. Stefanova, S. Basaia, M. P. van den Heuvel, T. Stojković, E. Canu, I. Stanković, V. Spica, M. Copetti, D. Gagliardi, V. S. Kostić, and M. Filippi. Structural brain connectome and cognitive impairment in parkinson disease. *Radiology*, 283(2):515–525, 2017. PMID: 27924721.
- [66] J. Gao, F. Liang, W. Fan, C. Wang, Y. Sun, and J. Han. On community outliers and their efficient detection in information networks. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, pages 813–822, New York, NY, USA, 2010. ACM.
- [67] T. Gartner, P. Flach, and S. Wrobel. On graph kernels: Hardness results and efficient alternatives. In *IN: CONFERENCE ON LEARNING THEORY*, pages 129–143, 2003.
- [68] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, Jun 2002.
- [69] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [70] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.
- [71] P. Goyal and E. Ferrara. Graph embedding techniques, applications, and performance: A survey. *CoRR*, abs/1705.02801, 2017.
- [72] R. M. Gray. *Entropy and Information Theory*. Springer-Verlag, Berlin, Heidelberg, 1990.
- [73] M. J. Greenacre. *Theory and Applications of Correspondence Analysis*. Academic Pr, 1984.
- [74] Greenlaw and D. Argnarsson. *Instructor's Resource Center for Graph Theory: Modeling, Applications, and Algorithms*. pearson, 2007.

- [75] A. Griffa, P. S. Baumann, C. Ferrari, K. Q. Do, P. Conus, J. P. Thiran, and P. Hagmann. Characterizing the connectome in schizophrenia with diffusion spectrum imaging. *Human Brain Mapping*, 36(1):354–366, 2015.
- [76] A. Griffa, P. S. Baumann, P. Klauser, E. Mullier, M. Cleusix, R. Jenni, M. P. van den Heuvel, K. Q. Do, P. Conus, and P. Hagmann. Brain connectivity alterations in early psychosis: from clinical to neuroimaging staging. *Translational Psychiatry*, 9(1):62, 2019.
- [77] A. Griffa, P. S. Baumann, J. P. Thiran, and P. Hagmann. Structural connectomics in brain diseases. *NeuroImage*, 80:515–526, 2013.
- [78] A. Griffa, B. Ricaud, K. Benzi, X. Bresson, A. Daducci, P. Vandergheynst, J.-P. Thiran, and P. Hagmann. Transient Networks of Spatio-temporal Connectivity Map Communication Pathways in Brain Functional Systems. *NeuroImage*, (April):1–13, 2017.
- [79] A. Grover and J. Leskovec. Node2vec: Scalable feature learning for networks. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD ’16, pages 855–864, New York, NY, USA, 2016. ACM.
- [80] L. Gutiérrez-Gómez and J.-C. Delvenne. Multi-hop assortativities for network classification. *Journal of Complex Networks*, page cny034, 2018.
- [81] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, Mar. 2003.
- [82] I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh. *Feature extraction: foundations and applications*, volume 207. Springer, 2008.
- [83] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1):389–422, Jan 2002.
- [84] P. Hagmann, L. Cammoun, X. Gigandet, R. Meuli, C. J. Honey, J. Van Welden, and O. Sporns. Mapping the structural core of human cerebral cortex. *PLoS Biology*, 6(7):1479–1493, 2008.
- [85] P. Hagmann, O. Sporns, N. Madan, L. Cammoun, R. Pienaar, V. J. Welden, R. Meuli, J.-P. Thiran, and P. E. Grant. White matter maturation reshapes structural connectivity in the late developing human brain.

- Proceedings of the National Academy of Sciences*, 107(44):19067–19072, 11 2010.
- [86] R. Haining. *Spatial Data Analysis: Theory and Practice*. Cambridge University Press, 2003.
 - [87] D. K. Hammond, P. Vandergheynst, and R. Gribonval. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129 – 150, 2011.
 - [88] Z. He and W. Yu. Stable feature selection for biomarker discovery. *Computational Biology and Chemistry*, 34(4):215 – 225, 2010.
 - [89] J. B. Heaton, N. G. Polson, and J. H. Witte. Deep learning for finance: deep portfolios. *Applied Stochastic Models in Business and Industry*, 33(1):3–12, 2017.
 - [90] C. Helma, R. D. King, S. Kramer, and A. Srinivasan. The predictive toxicology challenge 2000-2001. *Bioinformatics*, 17(1):107–108, Jan 2001.
 - [91] T. Horváth. *Cyclic Pattern Kernels Revisited*, pages 791–801. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.
 - [92] B. D. Hughes. *Random walks and random environments / Barry D. Hughes*. Clarendon Press Oxford ; New York, 1995.
 - [93] B. Iglewicz and . Hoaglin, David C. (David Caster). *How to detect and handle outliers*. Milwaukee, Wis. : ASQC Quality Press, 1993.
 - [94] M. Jenkinson, P. Bannister, M. Brady, and S. Smith. Improved optimization for the robust and accurate linear registration and motion correction of brain images. *NeuroImage*, 17(2):825–841, 2002.
 - [95] M. Jenkinson, C. F. Beckmann, T. E. Behrens, M. W. Woolrich, and S. M. Smith. Fsl. *NeuroImage*, 62(2):782–790, 2012.
 - [96] R. Kannan, S. Vempala, and A. Vetta. On clusterings: Good, bad and spectral. *J. ACM*, 51(3):497–515, May 2004.
 - [97] F. Karimi, M. Génois, C. Wagner, P. Singer, and M. Strohmaier. Homophily influences ranking of minorities in social networks. *Scientific Reports*, 8(1):11077, 2018.

- [98] H. Kashima, K. Tsuda, and A. Inokuchi. Marginalized kernels between labeled graphs. In *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), August 21-24, 2003, Washington, DC, USA*, pages 321–328, 2003.
- [99] J. G. Kemeny and J. L. Snell. *Finite Markov Chains: With a New Appendix "Generalization of a Fundamental Matrix" (Undergraduate Texts in Mathematics)*. Springer, 1983.
- [100] Y. Kim, S.-W. Son, and H. Jeong. Finding communities in directed networks. *Phys. Rev. E*, 81:016103, Jan 2010.
- [101] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *CoRR*, abs/1609.02907, 2016.
- [102] T. N. Kipf and M. Welling. Variational graph auto-encoders. *NIPS Workshop on Bayesian Deep Learning*, 2016.
- [103] D. Koutra, N. Shah, J. T. Vogelstein, B. Gallagher, and C. Faloutsos. Deltacon: Principled massive-graph similarity function with attribution. *ACM Trans. Knowl. Discov. Data*, 10(3):28:1–28:43, Feb. 2016.
- [104] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.
- [105] L. I. Kuncheva. A stability index for feature selection. In *Proceedings of the 25th Conference on Proceedings of the 25th IASTED International Multi-Conference: Artificial Intelligence and Applications, AIAP'07*, pages 390–395, Anaheim, CA, USA, 2007. ACTA Press.
- [106] R. Lambiotte, J.-C. Delvenne, and M. Barahona. Random walks, Markov processes and the multiscale modular organization of complex networks. *IEEE Transactions on Network Science and Engineering*, 1(2):76–90, Jul 2014.
- [107] A. N. Langville and C. D. Meyer. *Google's PageRank and Beyond: The Science of Search Engine Rankings*. Princeton University Press, Princeton, NJ, USA, 2006.

- [108] Q. Le and T. Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pages II-1188–II-1196. JMLR.org, 2014.
- [109] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998.
- [110] E. Lee, F. Karimi, H. Jo, M. Strohmaier, and C. Wagner. Homophily explains perception biases in social networks. *CoRR*, abs/1710.08601, 2017.
- [111] J. A. Lee, D. H. Peluffo-Ordóñez, and M. Verleysen. Multi-scale similarities in stochastic neighbour embedding: Reducing dimensionality while preserving both local and global structure. *Neurocomputing*, 169(Complete):246–261, 2015.
- [112] J. B. Lee, R. Rossi, and X. Kong. Graph classification using structural attention. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, pages 1666–1674, New York, NY, USA, 2018. ACM.
- [113] K. Lee, J. Caverlee, and S. Webb. Uncovering social spammers: Social honeypots + machine learning. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '10, pages 435–442, New York, NY, USA, 2010. ACM.
- [114] J. Li, H. Dani, X. Hu, and H. Liu. Radar: Residual analysis for anomaly detection in attributed networks. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 2152–2158, 2017.
- [115] M. W. Libbrecht and W. S. Noble. Machine learning applications in genetics and genomics. *Nature Reviews Genetics*, 16:321 EP –, May 2015. Review Article.
- [116] W. Lin, Y. Hu, and C. Tsai. Machine learning in financial crisis prediction: A survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(4):421–436, July 2012.

- [117] N. Liu, X. Huang, and X. Hu. Accelerated local anomaly detection via resolving attributed networks. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 2337–2343, 2017.
- [118] Q. Liu, M. Allamanis, M. Brockschmidt, and A. L. Gaunt. Constrained graph variational autoencoders for molecule design. In *Proceedings of the 32Nd International Conference on Neural Information Processing Systems, NIPS’18*, pages 7806–7815, USA, 2018. Curran Associates Inc.
- [119] L. Livi and A. Rizzi. The graph matching problem. *Pattern Anal. Appl.*, 16(3):253–283, Aug. 2013.
- [120] Y.-C. Lo, S. E. Rensi, W. Torng, and R. B. Altman. Machine learning in chemoinformatics and drug discovery. *Drug Discovery Today*, 23(8):1538 – 1546, 2018.
- [121] X. Lu, Y. Yang, F. Wu, M. Gao, Y. Xu, Y. Zhang, Y. Yao, X. Du, C. Li, L. Wu, X. Zhong, Y. Zhou, N. Fan, Y. Zheng, D. Xiong, H. Peng, J. Escudero, B. Huang, X. Li, Y. Ning, and K. Wu. Discriminative analysis of schizophrenia using support vector machine and recursive feature elimination on structural mri images. *Medicine (Baltimore)*, 95(30):e3973–e3973, Jul 2016. 27472673[pmid].
- [122] M.-E. Lynall, D. S. Bassett, R. Kerwin, P. J. McKenna, M. Kitzbichler, U. Muller, and E. Bullmore. Functional connectivity and brain networks in schizophrenia. *Journal of Neuroscience*, 30(28):9477–9487, 2010.
- [123] G. Ma, N. K. Ahmed, T. L. Willke, D. Sengupta, M. W. Cole, N. B. Turk-Browne, and P. S. Yu. Similarity learning with higher-order proximity for brain network analysis. *CoRR*, abs/1811.02662, 2018.
- [124] H. B. Mann and D. R. Whitney. On a test of whether one of two random variables is stochastically larger than the other. *Ann. Math. Statist.*, 18(1):50–60, 03 1947.
- [125] N. Masuda and P. Holme. Detecting sequences of system states in temporal networks. *Scientific Reports*, 9(1), 2019.
- [126] N. Masuda, M. A. Porter, and R. Lambiotte. Random walks and diffusion on networks. *Physics Reports*, 716-717:1 – 58, 2017. Random walks and diffusion on networks.

- [127] M. Meilă. Comparing clusterings-an information based distance. *Journal of Multivariate Analysis*, 98(5):873 – 895, 2007.
- [128] V. Metsis and et al. Spam filtering with naive bayes – which naive bayes? In *THIRD CONFERENCE ON EMAIL AND ANTI-SPAM (CEAS)*, 2006.
- [129] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS’13, pages 3111–3119, USA, 2013. Curran Associates Inc.
- [130] J. Moody. Race, school integration, and friendship segregation in america. *American Journal of Sociology*, 107(3):679–716, 2001.
- [131] E. Müller, P. I. Sánchez, Y. Mülle, and K. Böhm. Ranking outlier nodes in subspaces of attributed graphs. In *2013 IEEE 29th International Conference on Data Engineering Workshops (ICDEW)*, pages 216–222, April 2013.
- [132] L. Nanni, S. Ghidoni, and S. Brahmam. Handcrafted vs. non-handcrafted features for computer vision classification. *Pattern Recognition*, 71:158–172, 2017.
- [133] A. Narayanan, M. Chandramohan, L. Chen, Y. Liu, and S. Saminathan. subgraph2vec: Learning distributed representations of rooted sub-graphs from large graphs.
- [134] A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, and Y. Liu. graph2vec: Learning distributed representations of graphs.
- [135] P. Nemenyi. *Distribution-free multiple comparisons*. PhD thesis, Princeton University, 1963.
- [136] M. Newman. *Networks: An Introduction*. Oxford University Press, Inc., New York, NY, USA, 2010.
- [137] M. E. Newman. Assortative mixing in networks. *Phys. Rev. Lett.*, 89(20):208701, 2002.
- [138] M. E. J. Newman. The structure and function of complex networks. *SIAM REVIEW*, 45:167–256, 2003.

- [139] M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev. E*, 74:036104, Sep 2006.
- [140] M. E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, May 2006.
- [141] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 849–856. MIT Press, 2002.
- [142] M. Niepert, M. Ahmed, and K. Kutzkov. Learning convolutional neural networks for graphs. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML’16, pages 2014–2023. JMLR.org, 2016.
- [143] J. R. Norris. *Markov Chains*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 1997.
- [144] F. Orsini, D. Baracchi, and P. Frasconi. Shift aggregate extract networks. *CoRR*, abs/1703.05537, 2017.
- [145] L. Peel and A. Clauset. Detecting change points in the large-scale structure of evolving networks. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI’15, pages 2914–2920. AAAI Press, 2015.
- [146] L. Peel, D. B. Larremore, and A. Clauset. The ground truth about metadata and community detection in networks. *Science Advances*, 3(5), 2017.
- [147] D. Peluffo, J. Lee, M. Verleysen, J. L Rodríguez, and G. Castellanos-Dominguez. Unsupervised relevance analysis for feature extraction and selection a distance-based approach for feature relevance. In *3rd International Conference on Pattern Recognition Applications and Methods*, ICPRAM 2014, 01 2014.
- [148] B. Perozzi and L. Akoglu. Discovering communities and anomalies in attributed graphs: Interactive visual exploration and summarization. *ACM Trans. Knowl. Discov. Data*, 12(2):24:1–24:40, Jan. 2018.

- [149] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 701–710, New York, NY, USA, 2014. ACM.
- [150] N. Perraudin, B. Ricaud, D. I. Shuman, and P. Vandergheynst. Global and local uncertainty principles for signals on graphs. *APSIPA Transactions on Signal and Information Processing*, 7:e3, 2018.
- [151] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi. Defining and identifying communities in networks. *Proceedings of the National Academy of Sciences*, 101(9):2658–2663, 2004.
- [152] J. Ramon and T. Gartner. Expressivity versus efficiency of graph kernels. In *Proceedings of the First International Workshop on Mining Graphs, Trees and Sequences*, pages 65–74, 2003.
- [153] J. Richiardi, S. Achard, H. Bunke, and D. Van De Ville. Machine learning with brain graphs: Predictive modeling approaches for functional imaging in systems neuroscience. *IEEE Signal Processing Magazine*, 30(3):58–70, May 2013.
- [154] S. Ross. *Stochastic processes*. Wiley series in probability and statistics: Probability and statistics. Wiley, 1996.
- [155] Y. Saeys, I. Inza, and P. Larrañaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507–2517, 08 2007.
- [156] P. I. Sánchez, E. Müller, O. Irmeler, and K. Böhm. Local context selection for outlier ranking in graphs with multiple numeric node attributes. In *Proceedings of the 26th International Conference on Scientific and Statistical Database Management*, SSDBM '14, pages 16:1–16:12, New York, NY, USA, 2014. ACM.
- [157] M. T. Schaub, J.-C. Delvenne, S. N. Yaliraki, and M. Barahona. Markov dynamics as a zooming lens for multiscale community detection: Non clique-like communities and the field-of-view limit. *PLoS ONE*, 7(2):e32210, 2012.
- [158] M. Schmidt, G. Palm, and F. Schwenker. Spectral graph features for the classification of graphs and graph sequences. *Computational Statistics*, 29(1):65–80, 2014.

- [159] B. Schölkopf and A. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, USA, Dec. 2002.
- [160] I. Schomburg. Brenda, the enzyme database: updates and major new developments. *Nucleic Acids Research*, 32(90001), Jan 2004.
- [161] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, New York, NY, USA, 2004.
- [162] N. Shervashidze and K. M. Borgwardt. Fast subtree kernels on graphs. In *NIPS*, 2009.
- [163] N. Shervashidze, P. Schweitzer, E. J. van Leeuwen, K. Mehlhorn, and K. M. Borgwardt. Weisfeiler-lehman graph kernels. *J. Mach. Learn. Res.*, 12:2539–2561, Nov. 2011.
- [164] N. Shervashidze, S. Vishwanathan, T. Petri, K. Mehlhorn, and K. Borgwardt. Efficient graphlet kernels for large graph comparison. In *JMLR Workshop and Conference Proceedings Volume 5: AISTATS 2009*, pages 488–495, Cambridge, MA, USA, Apr. 2009. Max-Planck-Gesellschaft, MIT Press.
- [165] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, Aug. 2000.
- [166] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, May 2013.
- [167] D. I. Shuman, B. Ricaud, and P. Vandergheynst. Vertex-frequency analysis on graphs. *Applied and Computational Harmonic Analysis*, 40(2):260 – 291, 2016.
- [168] D. I. Shuman, P. Vandergheynst, D. Kressner, and P. Frossard. Distributed signal processing via chebyshev polynomial approximation. *IEEE Transactions on Signal and Information Processing over Networks*, 4(4):736–751, Dec 2018.
- [169] E. Simpson. Measurement of diversity. *Nature*, 163(4148):688, 1949.

- [170] S. W. Smith. *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Publishing, San Diego, CA, USA, 1997.
- [171] A. J. Smola and R. Kondor. *Kernels and Regularization on Graphs*, pages 144–158. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.
- [172] O. Sporns. The human connectome: a complex network. *Annals of the New York Academy of Sciences*, 1224(1):109–125, 2011.
- [173] O. Sporns, G. Tononi, and R. Kötter. The human connectome: A structural description of the human brain. *PLoS computational biology*, 1(4):e42, 9 2005.
- [174] A. Srinivasan, R. D. King, S. H. Muggleton, and M. J. E. Sternberg. The predictive toxicology evaluation challenge. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence - Volume 1, IJCAI'97*, pages 4–9, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.
- [175] J. Stehlé, N. Voirin, A. Barrat, C. Cattuto, L. Isella, J. Pinton, M. Quagiotto, W. Van den Broeck, C. Régis, B. Lina, and P. Vanhems. High-resolution measurements of face-to-face contact patterns in a primary school. *PLOS ONE*, 6(8):e23176, 08 2011.
- [176] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web, WWW '15*, pages 1067–1077, Republic and Canton of Geneva, Switzerland, 2015. International World Wide Web Conferences Steering Committee.
- [177] L. Tang and H. Liu. Leveraging social media networks for classification. *Data Mining and Knowledge Discovery*, 23(3):447–478, Nov 2011.
- [178] N. Tishby, F. C. Pereira, and W. Bialek. The information bottleneck method. In *Proc. of the 37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.
- [179] N. Tremblay and P. Borgnat. Graph wavelets for multiscale community mining. *IEEE Transactions on Signal Processing*, 62(20):5227–5239, Oct 2014.

- [180] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. In *Arxiv*, 2016.
- [181] L. van der Maaten and G. Hinton. Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 9(nov):2579–2605, 2008. Pagination: 27.
- [182] J. van Os, G. Kenis, and B. P. F. Rutten. The environment and schizophrenia. *Nature*, 468:203 EP –, Nov 2010. Perspective.
- [183] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning, ICML '08*, pages 1096–1103, New York, NY, USA, 2008. ACM.
- [184] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.*, 11:3371–3408, Dec. 2010.
- [185] N. Wale and G. Karypis. Comparison of descriptor spaces for chemical compound retrieval and classification. *Sixth International Conference on Data Mining (ICDM06)*, 2006.
- [186] A. H. Wang. Detecting spam bots in online social networking sites: A machine learning approach. In S. Foresti and S. Jajodia, editors, *Data and Applications Security and Privacy XXIV*, pages 335–342, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.
- [187] H. Wang, M. Tang, Y. Park, and C. E. Priebe. Locality statistics for anomaly detection in time series of graphs. *IEEE Transactions on Signal Processing*, 62(3):703–717, Feb 2014.
- [188] H. Wang, J. Wang, J. Wang, M. Zhao, W. Zhang, F. Zhang, X. Xie, and M. Guo. Graphgan: Graph representation learning with generative adversarial nets, 2018.
- [189] V. J. Wedeen, P. Hagmann, W. Y. I. Tseng, T. G. Reese, and R. M. Weisskoff. Mapping complex tissue architecture with diffusion spectrum magnetic resonance imaging. *Magnetic Resonance in Medicine*, 54(6):1377–1386, 2005.

- [190] R. C. Wilson, E. R. Hancock, and B. Luo. Pattern vectors from algebraic graph theory. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(7):1112–1124, July 2005.
- [191] R. C. Wilson and P. Zhu. A study of graph spectra for comparing graphs and trees. *Pattern Recogn.*, 41(9):2833–2841, Sept. 2008.
- [192] G. Wu, E. Y. Chang, and Z. Zhang. An analysis of transformation on non-positive semidefinite similarity matrix for kernel machines. In *Proceedings of the 22nd International Conference on Machine Learning*, 2005.
- [193] K. Xu, L. Wu, Z. Wang, Y. Feng, M. Witbrock, and V. Sheinin. Graph2seq: Graph to sequence learning with attention-based neural networks. *arXiv preprint arXiv:1804.00823*, 2018.
- [194] Y. Xu, K. Lin, S. Wang, L. Wang, C. Cai, C. Song, L. Lai, and J. Pei. Deep learning for molecular generation. *Future Medicinal Chemistry*, 11(6):567–597, 2019. PMID: 30698019.
- [195] P. Yanardag and S. Vishwanathan. Deep graph kernels. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 1365–1374, New York, NY, USA, 2015. ACM.
- [196] A. Yendiki, K. Koldewyn, S. Kakunoori, N. Kanwisher, and B. Fischl. Spurious group differences due to head motion in a diffusion mri study. *NeuroImage*, 88:79–90, Mar 2014. 24269273[pmid].
- [197] F. Zhang and E. R. Hancock. Graph spectral image smoothing using the heat kernel. *Pattern Recogn.*, 41(11):3328–3342, Nov. 2008.
- [198] T. Zhang, X. Zhuang, S. Pande, and W. Lee. Anomalous path detection with hardware support. In *Proceedings of the 2005 International Conference on Compilers, Architectures and Synthesis for Embedded Systems*, CASES '05, pages 43–54, New York, NY, USA, 2005. ACM.

- Adjacency matrix, 26
- Affected core, a-core, 127
- Anomaly detection on networks, 47
- Anomaly detection, Disney, 63
- Assortativity coefficient, 29
- Assortativity mixing, 28
- Autoencoder, 44

- Benchmark, chemoinformatics, 80
- Benchmark, social networks, 81
- Biomarker identification, 119

- Clustered autocovariance, 53
- community detection in networks, 29
- Complex networks, 17
- Concentration, 55
- Conditional entropy, 36
- Connectome, functional, 159
- Connectome, structural, 158

- Denoising autoencoder, 98

- Feature extraction, 38
- Feature selection, 37
- Filtering graph signals, 52

- Graph, 26
- Graph classification, 80, 110
- Graph clustering, 108
- Graph distances, 97
- Graph Fourier transform, 52
- Graph signals, 51
- Graph theory, 25
- Graph visualization, 103

- Iris dataset, 36

- Kernel, 40

- Laplacian matrix, 27

- Machine learning, 18, 36
- MADAN, 47
- Modularity, 29
- MRI, 157
- Multi-class classification, 42
- Multi-hop assortativities, 69
- Multi-hop assortativity, 74, 75
- Multilayer network, 110
- MWW test, 129

- Network embedding distances, 99
- Network embeddings, 93
- Network science, 17
- Node degree, 26
- Normalised mutual information, 108
- Normalized variation of information, 36

- Primary school network, 107
- Publications, 22

Random forest, 42
Random walk, continuous time, 32, 53
Random walk, discrete time, 31, 73
Random walks, 17
Recursive feature elimination SVM, 42
RFE-SVM, 42, 120, 123

Schizophrenia prediction, 117, 125
Stability, Kuncheva, 122
Stationary distribution, continuous, 32
Stationary distribution, discrete, 31
Supervised learning, 18, 38
Support vector machine, 38
SVM, 119

Unsupervised learning, 18, 43

Weighted adjacency matrix, 26