Università di Genova

DIPARTIMENTO DI
INFORMATICA, BIOINGEGNERIA,
ROBOTICA E INGEGNERIA DEI SISTEMI

Acknowledging the structured nature of real-world data with graphs embeddings and probabilistic inference methods

Davide Garbarino

Theses Series DIBRIS-TH-2022-42

Università di **Genova**

Dipartimento di Informatica, Bioingegneria,
Robotica ed Ingegneria dei Sistemi

Ph.D. Thesis in
Computer Science and Systems Engineering
Computer Science Curriculum

Acknowledging the structured nature of real-world data with graphs embeddings and probabilistic inference methods

by

Davide Garbarino

July, 2022

Ph.D. Thesis in Computer Science and Systems Engineering (S.S.D. INF/01)
Computer Science Curriculum
Dipartimento di Informatica, Bioingegneria,
Robotica ed Ingegneria dei Sistemi (DIBRIS)
Università di Genova

Candidate
Davide Garbarino
davide.garbarino@edu.unige.it

Title
Acknowledging the structured nature of real-world data with graphs embeddings and probabilistic inference methods

Advisors
Annalisa Barla
DIBRIS, Università di Genova
annalisa.barla@unige.it

External Reviewers
Derek Greene
University College Dublin, School of Computer Science, Science Centre - East Belfield Dublin 4
derek.greene@ucd.ie

Michele Starnini
ISI Foundation Institute for Scientific Interchange - Via Chisola 5 10126 Torino – Italy,
michele.starnini@gmail.com

Location
DIBRIS, Università di Genova
Via Dodecaneso, 35
I-16145 Genova, Italy

Submitted On
July 2022

Dedicated to V^2,
to whom I am inextricably bound.

Abstract

In the artificial intelligence community there is a growing consensus that real world data is naturally represented as graphs because they can easily incorporate complexity at several levels, *e.g.* hierarchies or time dependencies. In this context, this thesis studies two main branches for structured data.
In the first part we explore how state-of-the-art machine learning methods can be extended to graph modeled data provided that one is able to represent graphs in vector spaces. Such extensions can be applied to analyze several kinds of real-world data and tackle different problems. Here we study the following problems: *a*) understand the relational nature and evolution of websites which belong to different categories (e-commerce, academic (p.a.) and encyclopedic (forum)); *b*) model tennis players' scores based on different game surfaces and tournaments in order to predict matches results; *c*) analyze preterm-infants motion patterns able to characterize possible neuro degenerative disorders and *d*) build an academic collaboration recommender system able to model academic groups and individual research interest while suggesting possible researchers to connect with, topics of interest and representative publications to external users.
In the second part we focus on graph inference methods from data which present two main challenges: missing data and non-stationary time dependency. In particular, we study the problem of inferring Gaussian Graphical Models in the following settings: *a*) inference of Gaussian Graphical Models when data are missing or latent in the context of multiclass or temporal network inference and *b*) inference of time-varying Gaussian Graphical Models when data is multivariate and non-stationary. Such methods have a natural application in the composition of an optimized stock market portfolio.
Overall this work sheds light on how to acknowledge the intrinsic structure of data with the aim of building statistical models that are able to capture the actual complexity of the real world.

Publications

Part of materials and figures included in this thesis have previously appeared in the following publications:

JOURNAL PUBLICATIONS

1. *Garbarino, Davide, Giampaoli, Daniele, Cuneo, Marina, Paniati, Giorgia, Vian, Andrea, & Barla, Annalisa (SUBMITTED). Interactive recommendation in Academic Collaboration Networks. In the Special Section on "Emerging Trends and Advances in Graph-based Methods and Applications", IEEE Transactions on Emerging Topics in Computing.*

CONFERENCE PROCEEDINGS

1. *Garbarino, Davide, Moro, Matteo, Tacchino, Chiara, Moretti, Paolo, Casadio, Maura, Odone, Francesca,& Barla, Annalisa (2021, November). Attributed Graphettes-Based Preterm Infants Motion Analysis. In International Conference on Complex Networks and Their Applications (pp. 82-93). Springer, Cham.*

2. *Tozzo, Veronica, Ciech, Federico, Garbarino, Davide, & Verri, Alessandro (2021, August). Statistical Models Coupling Allows for Complex Local Multivariate Time Series Analysis. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining (pp. 1593-1603).*

3. *Tozzo, Veronica, Garbarino, Davide & Barla, Annalisa. Missing Values in Multiple Joint Inference of Gaussian Graphical Models. In Proceedings of the 10th International Conference on Probabilistic Graphical Models, PMLR 138:497-508, 2020.*

4. *Bayram, Firas, Garbarino, Davide, & Barla, Annalisa (2021, January). Predicting Tennis Match Outcomes with Network Analysis and Machine Learning. In International Conference on Current Trends in Theory and Practice of Informatics (pp. 505-518). Springer, Cham.*

5. *Garbarino, Davide, Tozzo, Veronica, Vian, Andrea, & Barla, Annalisa (2020, September). A robust method for statistical testing of empirical power-law distributions. In International Workshop on Algorithms and Models for the Web-Graph (pp. 145-157). Springer, Cham.*

Acknowledgements

I am extremely grateful to all the people that helped me during my PhD. I would like to thank my advisor, Annalisa Barla. She trusted me, supported me and, above all, providing me with comfort and friendship.
Special thanks to Veronica who guided me in my research, supported me and encouraged me at the same time. Thanks to Vanessa for always being by my side, ready to listen to me and help me in all ways that are possible for her. To Claudio for having shared a difficult and trying path from the beginning and for the precious silences we shared . To Margherita for the laughs we had together and for never letting me take myself too seriously.
To my brother Gabriele, the shoulder on which I lean for comfort and the dearest friend that life has given me. To my brother Simone for understanding my irony so deeply and for the love he gives me. To Cecilia, for allowing me to enter her world and learning so much. To Ilaria, my safe haven when I need to feel at home.

To my mum for everything.

Contents

List of Figures

List of Tables

1

Introduction

Real world complex systems (both natural ad artificial) are naturally described by *graphs*, that is data structures able to incorporate complexity at different levels, *e.g.* hierarchies or time dependencies. In general terms, a graph is simply a set of entities (*i.e.*, nodes), connected by a collection of interactions (*i.e.*, edges). For example, a social network might be represented as a graph in which nodes identify users and edges define mutual friendship relationships. Urban traffic networks model nodes as the intersections between roads and the edges mean that the road sections by which the two intersections can be connected directly (Figure 1). In the biological domain, transcriptional regulatory networks describe the regulatory interactions between genes: here, nodes correspond to individual genes and an edge is drawn from gene A to gene B if A positively or negatively regulates gene B.
Graph structures are a powerful tool that allows to focus on relationships between points (rather than the attributes of individual points), as well as model complex systems in its most general sense. Indeed, the same graph formalism can be used to represent social networks, interactions between drugs and proteins, the interactions between road intersections in a traffic network or the web pages on the World Wide Web to name just a few examples.

In the last decades, graph data quantity and quality considerably increased, which has led researchers to develop sophisticated and general machine learning analysis, representation and modeling techniques capable of exploiting the data to its almost full potential.
Before dealing in depth with machine learning methods for graphs, we give some introductory definitions.

Definition 1 *A homogeneous graph $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ is defined by a set of nodes $\boldsymbol{V} = \{v_1, \ldots, v_{|\boldsymbol{V}|}\}$ and a set of edges $\boldsymbol{E}$ between nodes. We denote an edge going from node $u \in \boldsymbol{V}$ to node $v \in \boldsymbol{V}$ as $(u, v) \in \boldsymbol{E}$. Graphs can be represented through the* adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{|\boldsymbol{V}| \times |\boldsymbol{V}|}$, $\boldsymbol{A} = (a_{i,j})_{i,j=1}^{|\boldsymbol{V}|}$, *where,*

$$a_{i,j} = \begin{cases} 1, & \text{if } (v_i, v_j) \in \boldsymbol{E} \\ 0, & \text{otherwise} \end{cases}. \tag{1}$$

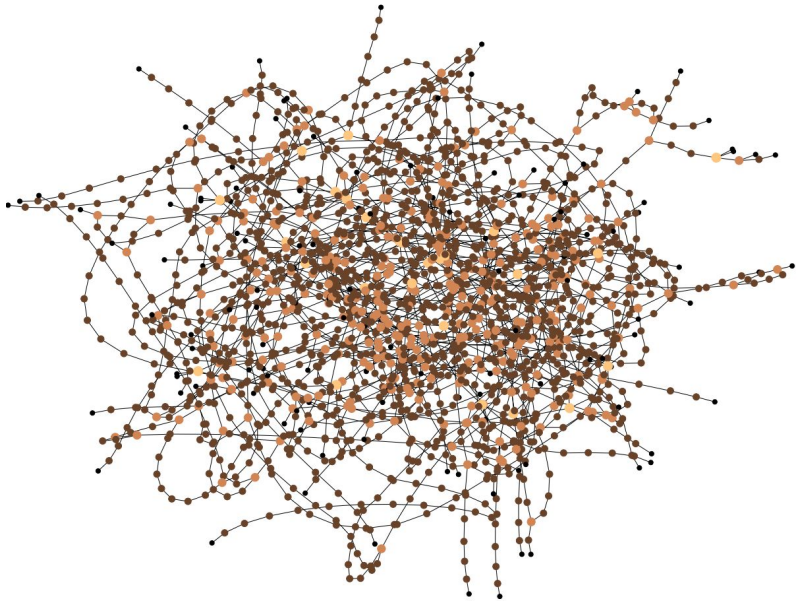


Figure 1: Luxembourg road network [218]. Nodes represent intersections between roads and the edges are connections between two intersections. Larger-sized and lighter-coloured nodes correspond to nodes with higher degree centrality values.

If the graph is undirected (i.e. *edge directions do not matter), the adjacency matrix* $\boldsymbol{A}$ *is symmetric, but if the graph is* directed, *meaning that edge directions matter,* $\boldsymbol{A}$ *is not necessarily symmetric.*

Beyond the distinction between undirected and directed graphs, we also consider graphs that have multiple types of nodes and edges.

Definition 2 *A* Heterogeneous Information Network *(HIN) is defined as a network* $\boldsymbol{G}$*=(*$\boldsymbol{V}$*,*$\boldsymbol{E}$*,*$\boldsymbol{T}$*) in which each node* $v \in \boldsymbol{V}$ *and each edge* $e \in \boldsymbol{E}$ *are associated with their* type functions $\mu(v) : \boldsymbol{V} \longrightarrow \boldsymbol{T_V}$ *and* $\lambda(e) : \boldsymbol{E} \longrightarrow \boldsymbol{T_E}$ *respectively.* $\boldsymbol{T_V}$ *and* $\boldsymbol{T_E}$ *denote the sets of object and relation types, where* $|\boldsymbol{T_V}| + |\boldsymbol{T_E}| > 2$.

Definition 3 A *Multilayer graphs is a network made up by multiple layers, each of which represents a given operation mode, social circle, or temporal instance. In a multilayer network every node is assumed to belong to every layer, and each layer corresponds to a unique relation, representing the intra-layer edge type for that layer. Moreover, inter-layer connect the same node across layers. Figure 2 describes a multilayer network, in which layers identify the temporal evolution of mutual friendship relationships in a social network.*

In many cases, nodes and/or edges in a graph have attributes or features information associated with (*e.g.*, keywords counts or topic distributions in networks semantically connecting documents retrieved from the Web). Most often these are node-level attributes that we represent using a real-valued matrix $\mathbf{X} \in \mathbb{R}^{d \times |\mathbf{V}|}$, where $d \in \mathbb{Z}_+$ is the dimension of each feature vector.

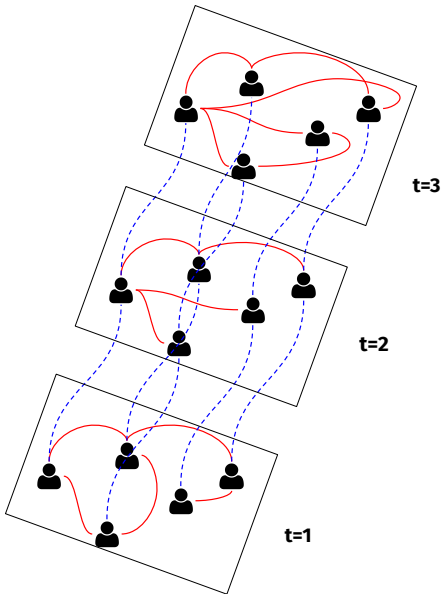


Figure 2: Schematic representation of a multilayer social network, where layers identify temporal instances of mutual friendship relationships (*i.e.* intra-layer red solid edges) between the same set of users in the social framework. Inter-layer blue dotted edges connect users to themselves across timestamps.

Machine Learning on Complex Networks

Machine learning models are statistical models that learn from data in order to solve particular tasks and are categorized according to the task they try to solve [122]. In particular, depending on the availability of labeled data, machine learning models are divided into the following categories:

- *Supervised learning models* use a training set to learn the desired output. This training dataset includes inputs and correct outputs, which allow the model to learn over time. The algorithm measures its accuracy through the loss function, adjusting until the error has been sufficiently minimized. Supervised learning models can be subdivided into two types of problems, *i.e.* classification and regression:
  - classification models accurately learn to assign test data into specific discrete categories.
  - regression models are used to understand the relationship between dependent and independent variables.
- *Unsupervised learning models* aim to analyze or cluster set of unlabeled data. These models discover hidden patterns or data clusters without the need for human intervention.
- *Semi-supervise learning models* are used when a small number of labeled examples and a large number of unlabeled examples are involved. Semi-supervised learning problems are challenging as neither supervised nor

unsupervised learning algorithms are able to make effective use of the mixtures of labeled and unlabeled data. As such, specific semi-supervised learning algorithms are required.

When dealing with graphs some additional attention must be taken into account, since the data structure tends to blur boundaries between usual machine learning categories. In the following sections we aim to describe main machine learning tasks and algorithms on graph data.

Node Classification

When dealing with large graphs, such as those that arise in the context of online social networks, a subset of nodes may be labeled. These labels can have several meanings ranging from demographic values to beliefs, or other characteristics of the nodes (users). A core problem is to use this information to extend the labeling so that all nodes are assigned a label (or labels). Such a problem is known as *node classification*.
Node classification is perhaps the most popular machine learning task on graph data, especially in recent years [259].
Examples of node classification beyond social networks for example include classifying the topic of documents based on hyperlink or citation graphs [139].
Node classification appears to be a direct variation of standard supervised classification, but actually they differ in one fundamental aspect: nodes in a graph are not independent and identically distributed (*i.i.d.*). Supervised machine learning models are usually built on the assumption that observed data are statistically independent of each other since, if not, we should also take dependencies into account during the modeling phase. As a guarantee of sufficient generalization capability for the machine learning model, we also assume that observed data are identically distributed. When dealing with graphs, node classification does not respect this assumption. Nodes of a graph are interconnected and most successful node classification algorithms explicitly leverage these connections.
Most recent node classification models are built upon node representation learning models which, in turn, seek to find similar representations to nodes who share overlapping or similar neighborhoods[84, 108, 201]. Therefore, when building node classification models, we are more interested in modelling node relationships than treating nodes as independent data points.

Link Prediction

Suppose we are given an incomplete graph $\mathbf{G} = (\mathbf{V}, \mathbf{E}_{\text{train}})$, meaning that we are provided with the full set of nodes $\mathbf{V}$ and an incomplete set of edges $\mathbf{E}_{\text{train}} \subseteq \mathbf{E}$, where $\mathbf{E}$ is the true underlying set of edges in the graph $\mathbf{G}$. The problem of inferring missing relationships (*i.e.* $\mathbf{E} \setminus \mathbf{E}_{\text{train}}$) among nodes is called *link prediction*. It is one of the most popular machine learning tasks with graph data and has countless real-world applications, including recommendation sys-

tems in e-commerce or social platforms [119, 155, 264] and reconstruction of protein-protein interaction (PPI) networks [148].
Link prediction complexity strongly depends on the typology of graph we deal with. Indeed, in homogeneous networks, characterized by single types of nodes and edges, graph heuristic based on neighborhood intersections can be leveraged in order predict the existence of a link between two nodes [160]. Nevertheless, when graphs are equipped with several types of nodes and edges, *e.g.* knowledge graphs, alternative techniques [84, 190], for instance based on metapaths, are required in order to accomplish the task.
Link prediction is a machine learning task on graph data that overcomes the dichotomy supervised-unsupervised learning. Indeed, there are cases in which link prediction can be faced as a completely supervised learning problem, specifically when only node attributes are exploited to infer the presence or absence of edges. In other cases, when node attributes are not available and we can only leverage graph topology, link prediction is closer to a semi-supervised learning problem since we look at all the graph to obtain suitable node and/or edge representations, whose thresholded similarity values can be used to predict missing edges [145].

Community detection

Many real-world complex networks exhibit a community structure, *i.e.* the organization of vertices in clusters, with many edges joining vertices of the same cluster and comparatively few edges joining vertices of different clusters. Such clusters, or communities, can be considered as fairly independent compartments of a graph, playing a similar role like, *e.g.*, the tissues or the organs in the human body. Community detection is an applied area in complex networks community [95], in which algorithms are defined in order to detect clusters of nodes on graphs. Such algorithms are completely unsupervised, differently from the previous ones. Real-world applications of community detection include uncovering functional modules in genetic networks [4], detecting communities of users in social platforms [33] and uncovering fraudulent groups of users in transaction networks [195].

Probabilistic Network Inference

The understanding of complex phenomena is a problem that arises in many applicative fields, such as finance, social science, medicine and biology [92, 125, 132]. Examples of complex phenomena are the evolution of a disease, the self regulation of the financial market, or, the change in social response to political decisions. All these phenomena can be looked at as systems composed of smaller entities that may or may not act independently. Often, the study of complex systems is fragmented in simpler tasks. One could look for the set of meaningful entities that are responsible for a specific state of the system (*e.g.*, identifying the genes responsible for the development of a specific cancer), or,

one could learn how to predict the behaviour of the system given the observations of its entities (*e.g.*, given a set of gene expressions predict the disease subtype affecting the patient). Nonetheless, these tasks are typically guided by prior knowledge and provide a simple and interpretable solution that, in turn, explains only a small portion of the phenomenon. Therefore, we could say that we are not understanding the system as a whole but we are simply describing some specific aspects of it. On the contrary, fully understanding a phenomenon entails a comprehension of the dynamic of interactions among entities as well as how these interactions relate to different statuses. Hence, in the presence of an explicit relationship between interacting entities and status, the understanding of the phenomenon can be simplified to the observing and learning over time how the entities that are part of the system contribute to a certain effect by interacting with each other.

The most suitable mathematical model for the abstract representation of entities and their interactions is a graph or network, that provides a compact representation of entities as nodes and their connections as edges. In the ideal case the graph model is known a priori, but often it needs to be inferred. The inference can be performed with a variety of different approaches. In this section we put ourselves in a machine learning setting: we observe the behaviour of (possibly a subset of) the entities within the system and we infer the best approximation of their connections under the form of a graph [100, 147]. Such approach is known as probabilistic network inference or graphical model selection.

Network inference can be performed through different strategies, typically based on different theoretical assumptions on the meaning of the edges. Here, we consider Markov Random Fields (MRFs) a set of statistical models that consider the connections between entities to describe conditional dependence. To this aim, entities are modelled by random variables that are assumed to follow a proper joint probability distribution.

The inference of a MRF on D variables becomes challenging when we are dealing with large scale data sets, *i.e.*, we have thousands of variables in play. Indeed, it consists in a combinatorial problem of identifying the correct network structure in a search space of possibly $2^{\frac{D(D-1)}{2}}$ edges. Thus, a reliable inference requires a large number of samples that are D-dimensional vectors of observations. Nonetheless, typically, the required sample size is not available, therefore the number of variables is much higher than the number of samples ($N \ll D$). The leading strategy to cope with this issue is to assume that just a reduced number of interactions are actually meaningful to the phenomenon under study and, therefore, constrain the problem and reduce the search space. In particular, we exploit regularised methods that impose a sparse prior on the problem [100, 171]. The sparse assumption eases the computational burden allowing us to find an approximate solution. At the same time, given the restricted set of resulting edges, it also improves interpretability of the graph. While being fundamental for identifiability, regularisation can be also leveraged to extend graphical models in order to consider more complex scenarios as multiple classes, longitudinal data, multi-level networks, latent variables

and many other possible conditions [54, 61, 74, 102, 109]. Throughout this thesis we will handle only methods for the inference of MRFs based on a sparse prior that recur to further regularisation strategies to cope with complex settings.

Regularised methods for the inference of complex MRFs have been proposed in the last few years in the context of continuous data (*i.e.*, the variables are assumed jointly Gaussian). In literature the so-called Gaussian Graphical Models (GGMs) have been considered in the presence of multi-class data [74, 109], temporal data [102, 112], multi-level networks [61], latent variables [54] and many others. Here, we mainly focus on temporal graphical models inferred from multivariate time-series under different settings as they allow to study an evolving system by modelling the underlying changes of the entities connections. We argue that considering the temporal component is fundamental in order to being truly able to understand a system. Indeed, as a system evolves the interactions among the variables of which is composed may change as well. Therefore, inferring its underlying structure in a unique steady state could be limiting for the detection of variability patterns. Note that, in reality there are systems for which the most suitable model is a unique structure that remains stable over time. Nonetheless, here, we want to focus on non-stationary systems whose understanding is bound to the observation of their evolution. This is particularly evident in some applications, such as biology, where the interest could be to understand the response of the system to perturbation [179].To this aim, [112] proposed a regularised extension of a method for the inference of stationary GGMs, the Graphical Lasso (GL) [100]. Such extension allows for the inference of networks at discrete time points connected through a specific dynamical behaviour. This method assumes Markovianity, *i.e.*, each time point is dependent on the previous one. To force such dependency, it employs a temporal consistency function that yields network structures close in time to be similar. This method was shown to improve inference with respect to static methods as it allows to consider the global evolution of the system thus providing a more sound and stable inference of the underlying network. Moreover, it allows to study evolving patterns that are impossible to detect otherwise. Nonetheless, while being extremely powerful, we point out two aspects as drawbacks of such model:

- It does not consider the presence of missing data [154] which influence how the observable entities are perceived and, hence, which interactions are learned [64];
- Data are assumed to be identically distributed across time points [112], meaning that the time series is stationary. Nonetheless, many real world examples, such as financial markets, involve variables that change over time showing possible trends.

The first aspect may perturb the final results and thus, entails the need of inserting missing data assumptions in the inference process to avoid misrepresentations [172]. The second aspect does not allow us to use standard methods

for the inference of time-varying networks and therefore raises the problem of clustering equally distributed observations before inferring probabilistic relationships among variables.

Contributions

In this thesis, we propose major contributions within the two macro areas described above, *i.e.* extending state-of-the-art machine learning methods to graph modeled data and filling the aforementioned gaps in probabilistic network inference methods. More specifically, we propose four adaptations of machine learning algorithms to graph-modelled data, *i.e.*:

- we leverage topological graph representations to understand the relational nature and evolution of web-sites which belong to different categories (e-commerce, academic (p.a.) and encyclopedic (forum));
- we define network-based features able to model tennis players scores on different game surfaces and tournaments in order to predict matches results;
- we leverage probabilistic models based on network statistics to suitably represent preterm-infants motion patterns in order characterize possible neuro degenerative disorders;
- we leverage recently defined heterogeneous graph neural networks to build an interactive academic collaboration recommender system. Our aim is to model academic groups and individual research interests while suggesting possible researchers to connect with, topics of interest and representative publications to external users.

Furthermore, we propose two inference method approaches within the network inference framework that attempt to overcome issues related with real data. Specifically,

- we propose possible extensions of the temporal network inference with Gaussian assumption in the case of missing data which may either present missing random values or variables that are consistently never measured, and that we define as latent. To solve these two problems we devised two different strategies, one builds on the Expectation Maximisation method and the second on the marginalisation of the latent variables effect. Finally, we show a case in which by exploiting partial prior knowledge on the latent variables, we can obtain results that go in the direction of multi-level networks;
- we provide a general coupled statistical model for the inference of graphs that is flexible to diverse consistency types and possible non-Markovian dependencies. Such method infers a graphical model from non-stationary multivariate time series under complex temporal dependency patterns.

Assuming that we do not know such patterns a priori, we also provide automatic identification techniques.

Overall this work sheds light on how to acknowledge the intrinsic structure of data with the aim of building statistical models that are able to capture the actual complexity of the real world.

Outline

The present thesis is structured in two main parts.

1. In the first one, we provide a comprehensive description of the state-of-the-art methods for complex network representations, mainly used as inputs to machine learning models on graph data. We list and describe some of the most studied and used network measurements in Chapter 2 . In Chapter 3 we describe random walk- and deep learning-based methods for learning representations of homogeneous and heterogeneous graphs. In Chapter 4 we present the original contribution of this thesis. In particular, in Section 4.1 we provide an extension of a state-of-the-art method for testing and questioning the overall ubiquitous presence of scale-free networks in real-world complex systems. In Section 4.2 and Section 4.3, we describe applications of classical networks representation learning in two different scenarios, *i.e.* tennis tournaments and preterm infants motion analysis. Finally, in Section 4.4, we propose a preliminary analysis of a heterogeneous information network, *i.e.* the collaboration network of the MaLGa academic research group.

2. In the second part we describe the context and contributions about regularized network inference. Chapter 5 presents the steady-state regularised inference methods under the Gaussian distribution assumptions. Chapter 6 focuses on contributions on the concept of missing data in Gaussian Graphical Models, both at random or latent, and on the study of temporal Gaussian Graphical Models when data is non-stationary.

Notation

Unless explicitly specified, we denote with bold upper-case letters $\mathbf{G}, \mathbf{V}, \mathbf{E}$ and $\mathbf{A}$ graphs, sets of vertices and edges, and adjacency matrices respectively. With denote probability distributions as p, unless otherwise specified. Uni-dimensional vectors are represented as with bold lwer-case letters x and with upper case letters X 2-dimensional matrices.
The entries of vectors and matrices are denoted by $x[i]$ (or equivalently x_i) and $X[ij]$ (or equivalently $X[i,j]$ and $X_{i,j}$) respectively. When we want to select an entire dimension we put a colon, *e.g.*, if we have a 2- dimensional matrix and we want to take the i-th row we write $X[i,:]$. If we want to select all but one row we will write $X[-i,:]$. Given a set of indices $\mathbb{I}_A = \{1,\dots,j\}$ we denote the squared sub-matrix obtained by selecting the corresponding rows and columns as $X[A]$. Similarly we denote the sub-matrix obtained by selecting the rows in the set $\mathbb{I}_A$ and the columns in the set $\mathbb{I}_B$ by $X[AB]$. We will denote the cardinality of the set $\mathbb{I}_B$ with $|B|$. With $\mathbb{S}^D_{++}$ we denote the cone of positive definite matrices, similarly $\mathbb{S}^D_{+}$ denotes the cone of positive semi-definite matrices. It is equivalent to say that $X\mathbb{S}^D_{++}$ $X \succ 0$, similarly $X\mathbb{S}^D_{+}$ is equal to $X \succeq 0$. With $\langle \cdot, \cdot \rangle$i we denote the scalar product between two vectors.

PART I

Representation of Complex Networks

This part of the document covers a detailed presentation of the state-of-the-art methods for complex network representations. First, we describe some of the most studied and used network measurements. Then, we describe random walk- and deep learning-based methods for representing homogeneous and heterogeneous graphs. We conclude the first part of thesis presenting original contributions about applications of machine learning algorithms to real-world complex data.

2

Topological Features Representation of Complex Networks

2.1 Introduction

Nowadays, complex networks are pervasive in real world applications ranging from sociology [225] to biology [227]. One of the main reasons behind complex networks popularity is their flexibility and generality for representing virtually any natural structure.
Several investigations in complex network involve the representation of the structure of interest as a network (interchangeably called *graph*), followed by an analysis of the topological features of the obtained representation performed in terms of a set of informative measurements.
Such analysis is usually aimed to accomplish the following three tasks:

- the topological characterization of the studied structures [71];
- the identification of different categories of (sub)structures or nodes, which is directly related to the area of *machine learning* [17, 93];
- the selection of a plausible random graph model able to describe the empirical structure and its evolution [17, 135].

All of the above mentioned tasks imply the same crucial question of how to choose the most appropriate measurements able to best either describe or discriminate network topologies. Such a choice should reflect the specific application, nevertheless no formal procedure is available for identifying the best measurements. There is an huge set of topological measurements, showing high level of correlation and then redundancy. Statistical approaches to decorrelation (e.g., *principal component analysis* [1]) can help select and enhance measurements, but are not guaranteed to produce optimal results. Furthermore, one has to rely on her knowledge of the problem and available measurements in order to select a suitable set of features to be considered. For such reasons, it is of paramount importance to have a good knowledge not only of the most

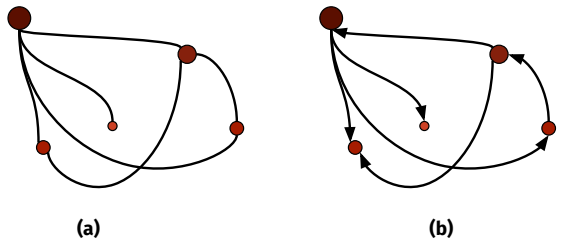


Figure 3: Examples of undirected (a) and directed graphs (b).

representative measurements, but also of their respective properties and interpretation.

OUTLINE In the remainder of the chapter, we describe some important node-level features and statistics, and discuss how these can be generalized to graph-level statistics. The main goal of the chapter is to introduce various heuristics, which are often used as features in traditional machine learning algorithms applied to graphs.

2.2 Topological Measurements

In a given network $\mathbf{G}$, we denote $\mathbf{V}$ the set of *nodes* (or *vertices*), $\mathbf{E}$ the set of *edges* (or *links*) and $\mathbf{A} \in \mathbb{R}^{|V|\times|V|}$ the binary adjacency matrix. The number of nodes is $n = |V|$ and the number of edges is $m = |E|$.
Nodes in $\mathbf{V}$ are denoted v_i for $i = 1, \ldots, n$ and edges are denoted as the pair (v_i, v_j) for $i, j = 1, \ldots, n$.
For undirected graphs (Figure 3 (a)), two vertices v_i and v_j are said to be *adjacent* or *neighbors* if the corresponding entry in the adjacency matrix $a_{ij} = a_{ji} = 1$.
For directed graphs (Figure 3 (b)), the corresponding concepts are those of *predecessor* and *successor*: if $a_{ij} = 1$ then v_i is a predecessor of v_j and v_j is a successor of v_i. The *neighborhood* of a vertex v_i, denoted $\mathcal{N}(v_i)$, corresponds to the set of nodes adjacent to v_i.

2.2.1 *Node-level Statistics*

We now describe some notable topological measurements of complex networks that could be used as features in a machine learning model for either node classification, node clustering or regression.

DEGREE This nodal measure d_{v_i}, *i.e.* the degree of node v_i, corresponds to the number of edges attached to v_i, equivalently defined as the cardinality of

$\mathcal{N}(v_i)$. For undirected networks, the degree d_{v_i} can be formally defined using the adjacency matrix:

$$d_{v_i} = \sum_{v_j \in \mathbf{V}} a_{ij} = \sum_{v_j \in \mathbf{V}} a_{ji}. \tag{2}$$

For directed networks, we need to distinguish between *incoming* and *outgoing links* and then between *in-degree* ($d^{in}_{v_i}$), and *out-degree* ($d^{out}_{v_i}$) of a node v_i:

$$d^{in}_{v_i} = \sum_{v_j \in \mathbf{V}} a_{ji}, \tag{3}$$

$$d^{out}_{v_i} = \sum_{v_j \in \mathbf{V}} a_{ij}. \tag{4}$$

In this case, we define the *total degree*, or more in general *degree*, of a node v_i as $d_{v_i} = d^{in}_{v_i} + d^{out}_{v_i}$.
In general, the degree of a node is an essential statistic to consider, and it is often one of the most informative features in traditional machine learning models applied to node-level tasks [35] or generative model selection [182].

DISTANCE A *path* between two non-adjacent nodes v_i and v_j is defined as a sequence of edges $p_{v_i,v_j} = (v_i, v_{k_1}), \ldots, (v_{k_l}, v_j)$, with $l \geq 1$, connecting v_i and v_j. Then, a *shortest path* between two non-adjacent nodes v_i and v_j is defined as a path $sp_{i,j}$ which has minimum *length*, *i.e.* contains a minimum number of edges. The *distance* $d(v_i, v_j)$ between two nodes v_i and v_j corresponds to the length of the shortest path between them [106].

NODE CENTRALITY Node degree simply measures how many neighbors a node has, but this is not necessarily sufficient to measure the importance of a node in a graph.
In many cases we can benefit from additional measures of node importance. To obtain a more powerful measure of importance, we can consider various measures of what is known as *node centrality*, which can form useful features in a wide variety of tasks.

- the *betweenness centrality* of a node v_i, $b_c(v_i)$, measures the extent to which v_i lies between other nodes in the network and can be computed as the percentage of shortest paths that pass through the node. The betweenness centrality of node v_i can be computed as

$$b_c(v_i) = \sum_{v_j, v_k \in \mathbf{V} \setminus \{v_i\}} \frac{\sigma_{v_j,v_k}(v_i)}{\sigma_{v_j,v_k}}, \tag{5}$$

  where $\sigma_{v_j,v_k}(v_i)$ and σ_{v_i,v_j} denote the number of shortest paths between nodes v_j and v_k passing through node v_i and the number of shortest

paths between nodes v_j and v_k respectively.
Nodes with high betweenness centrality occupy critical roles in the network topology, since they usually have a network position that allow them to work as an interface between tightly-knit groups.

- The nodal measure *closeness centrality* of v_i, $c_c(v_i)$, is the inverse of the sum of distances between the node of interest v_i and all the other nodes in the network $v_j \in \mathbf{V}\backslash\{v_i\}$. It quantifies how close a node is from the rest of the network, in average:

$$c_c(v_i) = \frac{1}{\sum_{v_j \in \mathbf{V}\backslash\{v_i\}} d(v_i, v_j)}. \tag{6}$$

CLUSTERING COEFFICIENT One way to characterize the presence of loops of order three in a node neighborhood $\mathcal{N}(i)$ in an undirected network is through the *clustering coefficient* [256].
Let $d(v_i)$ be the degree of node v_i, meaning the cardinality of $\mathcal{N}(v_i)$, and l_{v_i} the number of connected nodes in $\mathcal{N}(v_i)$, then the clustering coefficient of node v_i is defined as:

$$C_{v_i} = \frac{2l_{v_i}}{d(v_i)(d(v_i)-1)}, \tag{7}$$

where $\frac{d(v_i)(d(v_i)-1)}{2}$ represents the total number of possible existing links among the neighbors of v_i. As one can easily see, the clustering coefficient C_{v_i} measures the tendency of a node neighbors to be connected to each other.
A general definition of clustering coefficient in directed networks is given in following paragraphs, where we introduce the notion of *graphlet* which allows us to generalize the notion of degree and clustering coefficient.

GRAPHLET AND GRAPHETTE DEGREE VECTORS Graphlets [207], are small, connected, non-isomorphic, induced subgraphs of a larger graph $\mathbf{G}$. Graphlets generalize the notion of degree. Indeed, given a node v_i of an undirected network $\mathbf{G}$, the degree measures the number of edges attached to v_i. Note that an edge is the only graphlet with two nodes; henceforth, we call this graphlet G_0 (see Figure 4). Note that we can apply the same measurement to other graphlets as well as G_0. Thus, in addition to applying this measurement to an edge, *i.e.* graphlet G_0, we apply it to the 29 graphlets G_1, G_2,...,G_{29} presented in Figure 2 as well. When computing such occurrences it is important to take into consideration topological issues deriving from the exact position of the node in the graphlet.
To better understand such phenomenon, we give the following definition:
An *isomorphism* ψ from graph $\mathbf{G}$ to graph $\mathbf{H}$ is a bijection from $\mathbf{V}$(G) to $\mathbf{V}$(H) such that $(v, w) \in \mathbf{E}(\mathbf{G})$ if and only if $(\psi(v), \psi(w)) \in \mathbf{E}(\mathbf{H})$. An automorphism is an isomorphism from a graph $\mathbf{G}$ to itself. The set of automorphisms of a graph G equipped with the function composition operation forms a group denoted $Aut(\mathbf{G})$. Then, for every $v \in \mathbf{V}(G)$, we can define the *automorphism orbit* of v as $\pi(v) = \{w \in \mathbf{V}(\mathbf{G}) \mid w = \psi(v) \text{ for some } \psi \in Aut(\mathbf{G})\}$. In Figure

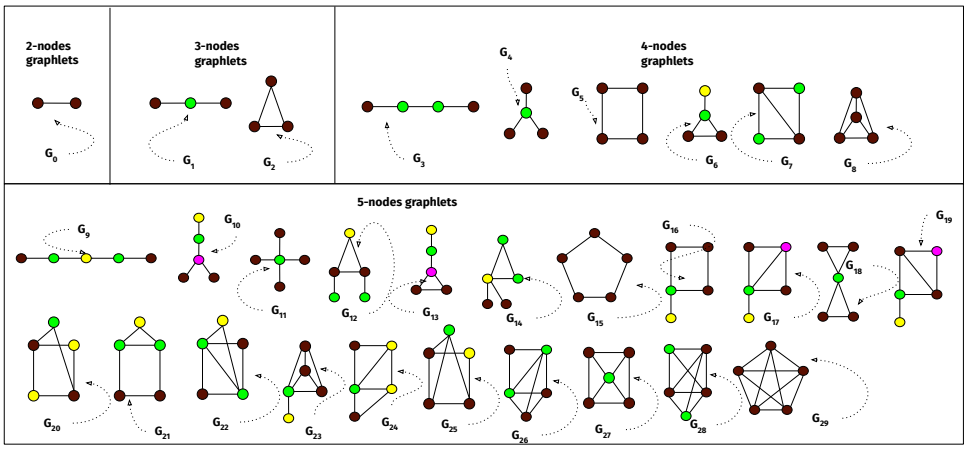


Figure 4: Automorphism orbits for the thirty 2, 3, 4, and 5-node undirected graphlets. In a graphlet $\mathbf{G}_i$, nodes belonging to the same orbit are of the same colour.

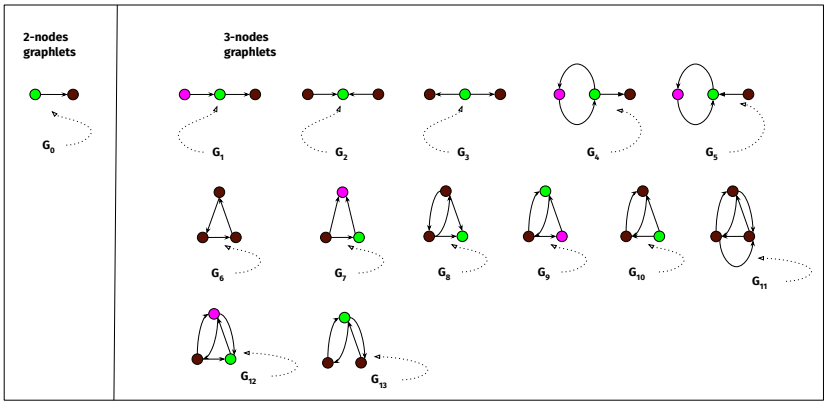


Figure 5: Automorphism orbits for the 14 2 and 3-node directed graphlets. In a graphlet $\mathbf{G}_i$, nodes belonging to the same orbit are of the same colour.

4 are depicted undirected graphlets $G_0, \ldots, G_{29}$ decomposed into their automorphism orbits (identified by the node colours). Hence, for every node v in a network $\mathbf{G}$ we can count how many times it appears in each of the 72 orbits thus obtaining a 72-dimensional representation of v. Such representation is called *graphlet degree vector* (GDV)[206]. GDV generalizes not only the notion of node degree but also the notion of clustering coefficient (*i.e.*, the normalized occurrences of graphlet G_2).

Directed graphlets[221] are defined similarly to undirected graphlets but we need to take into consideration edge orientations. This fact has the consequence to considerably increase the number of graphlets and corresponding automorphism orbits (see Figure 5), making the algorithms for calculating the occurrences of such substructures more complex [14, 221].
In literature, several algorithms for enumerating and counting node orbits in

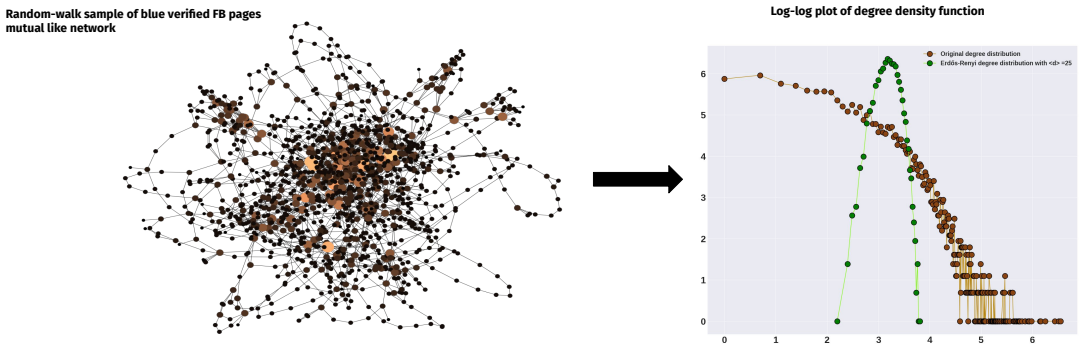


Figure 6: The left panel shows a random-walk sample of a large real-world network. Nodes represent blue verified Facebook pages [218] and edges are mutual likes among them. Light coloured and big-sized nodes correspond to high degree nodes. On the contrary, dark coloured and small-sized nodes have few connections. On the right panel, the log − log plot of the original degree distribution is compared to the log − log plot of the degree distribution of an Erdős-Renyi random graph instance. The original network and the model realization share the number of nodes and the average degree.

directed and undirected networks are defined[15, 129, 143, 221]. Typically these algorithms are based on the simple observation that more complex graphlets are built upon the combination of simpler graphlets (this is also the reason why usually only graphlets of size up to 6 are considered) and use combinatorial equations to connect their occurrences.
Recent generalizations of the notion of undirected graphlets, *i.e. graphettes* [118], obtained by considering also disconnected induced subgraphs allow to speed up the computation of orbit frequencies in undirected networks while also allowing to extend the analysis to graphettes of size up to 8.

2.2.2 *Graph-level Statistics*

In the previous section we reviewed some topological measurements at the node-level. However there are cases in which we need to classify or cluster structures rather than nodes (*e.g.* the classification of microbiome trees in a case-control study). In this scenario graph-level statistics are best suited to accomplish the task.
In the next paragraphs we describe some notable graph measurements which often are defined as an average of node-level or random walk-based statistics.

DEGREE AND GRAPHLET DEGREE DISTRIBUTION (GDD) Consider an undirected network $\mathbf{G}$. The *degree distribution* measures, for each natural number k, the number of nodes of degree k. In other words, for each value of k, it gives

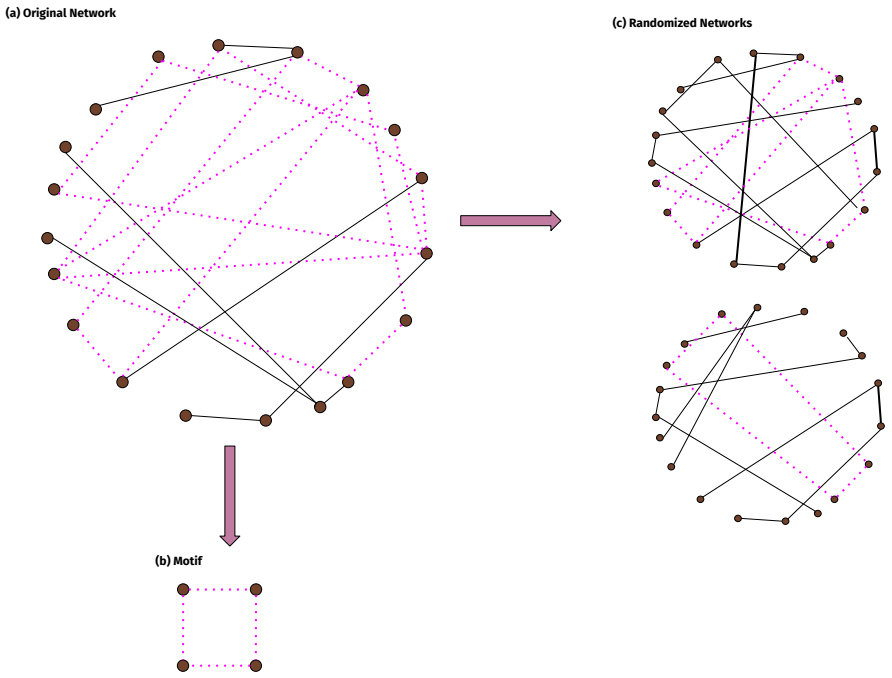


Figure 7: Example of motifs in a toy network. Looking at the original network (a) it is clear that the "squared" subgraph (b) (G_5 in Figure 4) is over represented in the original network w.r.t. randomized versions of it (c).

the number of nodes attached to k graphlets G_0 (Figure 4). Similarly, for every graphlet G_i with $i \in \{1, \ldots, 29\}$, we can count the number of nodes touching G_i at a node belonging to a particular orbit. Thus, we obtain 73 distributions analogous to the degree distribution, which are called the *graphlet degree distribution* (GDD).
Analogously, we can define the GDD for directed networks, whose in- and out-degree distributions correspond to the first two distributions in the GDD.

It is empirically observed that most real-world networks exhibit heavy tails in the (in- and out-) degree distribution [21] (Section 4.1). More specifically, real-world networks are often claimed to be *scale free* [23], meaning that the degree distribution follows a power-law [16], a pattern with broad implications for the structure (*e.g.* emergence of hubs [22, 137]) and dynamics (*e.g.* robustness to cascading failures [25, 85]) of complex systems (see Figure 6 for a real-world example). Testing if the tail of a real-world network degree distribution is plausibly approximable through a discrete power-law distribution is often used as a primary test in generative random graph model selection [182]. Indeed, such an observation would allow to exclude as likely generative models, for instance, Erdős-Renyi [87] random graph models whose degree distribution follows a Poisson distribution, as shown in Figure 6.

MOTIFMOTIFS CONCENTRATIONS CONCENTRATIONS *Network motifs* are recurring, significant patterns of interconnections [176]. Network motifs are substructures that recur much more frequently in a real network than in an ensemble of randomized networks, as shown in Figure 7. For this reason, motifs detection strongly depends on the assumption of a plausible generative random graph model whose real network is a realization. Indeed, such a random graph model allows us to generate randomized versions of the real network $\mathbf{G}$ and compare a specific detected motif concentration in $\mathbf{G}$ to the average concentration of such a motif in randomized graphs.
Exact methods for computing motif concentrations in undirected and directed networks are analogous to the graphlets enumeration methods. However, only for undirected networks, random walk-based algorithms to approximate motif concentrations in large complex networks have been defined in literature [36, 59, 116]. Such methods rely on suitably defined generalized random walks able to sample frequent subgraphs and approximate their concentration in the original network through the average of such subgraphs concentrations in the sampled graphs. One of the main assumption on which random walk-based approximation algorithms rely is the irreducibility [215] of the Markov Chain associated to the random walk. This is true for any random walk defined on a connected undirected graph, but, in general, it is not valid for Markov Chains associated to random walks defined on connected directed graphs.

AVERAGE SHORTEST PATH LENGTH AND DIAMETER The average shortest path length $asp(\mathbf{G})$ is defined as the normalized average distance among all connected pairs of nodes. Equivalently:

$$asp(\mathbf{G}) = \frac{1}{n(n-1)} \sum_{v_i, v_j \in \mathbf{V}, i \neq j} d(v_i, v_j), \tag{8}$$

where $d(v_i, v_j)$ is the distance between nodes v_i and v_j and is set to 0 if v_i and v_j are not connected in $\mathbf{G}$.
The behavior of the average shortest path length as a function of the number of nodes n of a random graph model indicates whether the model shows the *small-world property* [256]. More specifically, if $asp(\mathbf{G})$ scales as $O(\ln(n))$ then $\mathbf{G}$ can be plausibly considered as a realization from a *small-world random graph model* [187].

The *diameter* $D(\mathbf{G})$ of a connected network $\mathbf{G}$ is defined as the longest of all the calculated shortest paths in $\mathbf{G}$. Equivalently,

$$D(\mathbf{G}) = \max\{d(v_i, v_j) \mid v_i, v_j \in \mathbf{V}, i \neq j, v_i \text{ and } v_j \text{ are connected}\}. \tag{9}$$

AVERAGE CLUSTERING COFFICIENT For undirected networks, the *average clustering coefficient*, $C(\mathbf{GF})$ is defined as the average of each node clustering coefficient C_{v_i}, for $v_i \in \mathbf{V}$. Equivalently,

$$C(\mathbf{G}) = \frac{1}{n} \sum_{v_i \in \mathbf{V}} C_{v_i}. \tag{10}$$

High values of such measure, similarly to small values of the average shortest path length, is an indicator of the emergence of the small-world property [175].

2.3 Conclusion

In the previous sections, we presented a number of traditional approaches to learning over graphs. We saw how graph statistics can extract feature information for machine learning tasks.
However, the node and graph-level statistics are sub-optimal features due to the fact that they require handcrafted statistics and measures. As we observed, these hand-engineered features can be hard even to approximate, leading to a time-consuming and expensive process.
In the following chapters we focus on an alternative approach to learning over graphs: *graph representation learning*. Instead of extracting hand-engineered features, we will seek to learn representations that encode structural information about the network.

3

Node Representation Learning

In this chapter of the thesis we focus on methods for learning *node embeddings*, which allow to encode nodes as low-dimensional vectors that incorporate both information on their topological roles in the graph and the geometric structure of their neighborhoods. In practice, we want to learn a low-dimensional latent space into which geometrically near projected nodes correspond to proximal nodes in the original graph.

3.1 Introduction

Traditional machine learning approaches on graphs rely on precomputed heuristics to extract features encoding topological information (see Chapter 2). Nevertheless, the last decades have seen an increasing interest in approaches that *learn* to encode network structure and node/link semantics into low-dimensional embeddings. A schematic representation of node embeddings learning is presented in Figure 8.
Let $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ a graph and $\mathbf{V}$ and $\mathbf{E}$ represent nodes and edges of the graph respectively. A node embedding method maps nodes into a low-dimensional latent space ($\mathbb{R}^d$) such that $d \ll |V|$.

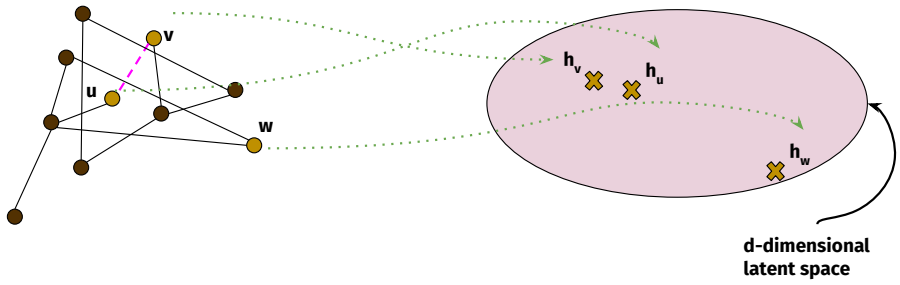


Figure 8: Schematic description of nodes representation learning on a toy network. Neighbors in the original graph are mapped in geometrically close points in the learned d-dimensional latent space.

Node embedding methods preserve different properties of nodes and edges in graphs such as node proximities. We define first-order and second-order proximities since higher order proximities can be similarly obtained.
Nodes that are connected by an edge have a *first-order proximity*. The *second-order proximity* of two nodes involve the degree of overlap of their neighborhoods, meaning that nodes sharing more neighbors are considered to be more similar to each other.

Node embedding methods can be categorized into four categories: *factorization based* [5, 51, 193], *random walk based*, *deep learning based* and *graph neural networks* (GNN) *based* methods. Below, we review only random walk based and GNN based methods while discussing their extensions to heterogeneous graphs.

3.2 Random Walk based Methods

The main concept that random walk based methods utilize is generating random walks for each node in the graph to capture the structure of the graph and output similar node embedding vectors for nodes that occur in the same random walks.
A random walk is defined as a sequence of nodes $v_1, \ldots, v_l$ that starts at node v_0, $(v_i, v_{i+1}) \in \mathbf{E}$ and l is the length of the walk. Given v_i in the sequence, next node v_{i+1} is chosen based on some probability distribution. In the following sections we describe details about two random walk based embeddings, *i.e.* DeepWalk [201] and Node2vec [108].

3.2.1 *DeepWalk and Node2vec*

DeepWalk [201] and Node2vec [108] are methods based on the notable word2vec embedding [174] which is frequently used in text mining tasks. Here, the assumption is that that words co-occurring in the same sentence several times have a similar meaning. Node2vec and DeepWalk translate this assumption into graph data by assuming that nodes co-occurring in many random walks have to be similar. Therefore, as a consequence such methods generate similar node embedding vectors for neighboring nodes.
Both methods leverage on a similar algorithm, which is composed by two distinct parts:

1. *random walks generation*. DeepWalk selects the next node in the random walk uniformly at random from the neighbors of the previous node. Differently, Node2vec defines a biased scheme for sampling consecutive nodes in the random walk. More precisely, given the random walk $v_0, v_1, \ldots, v_i$ we select v_{i+1} in the sequence based on the following probability distribution

$$p(v_i|v_{i+1}) = \begin{cases} \frac{\alpha_{v_i v_{i+1}}}{Z}, & \text{if } (v_i, v_{i+1}) \in \mathbf{E} \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

where Z is a normalization constant and $\alpha_{v_i v_{i+1}}$ is defined as

$$\alpha_{v_i v_{i+1}} = \begin{cases} \frac{1}{p}, & \text{if } d(v_{i-1}, v_{i+1}) = 0 \\ 1, & \text{if } d(v_{i-1}, v_{i+1}) = 1 \\ \frac{1}{q}, & \text{if } d(v_{i-1}, v_{i+1}) = 2 \end{cases} \tag{12}$$

where $d(v_{i-1}, v_{i+1})$ is the distance between nodes v_{i-1} and v_{i+1} and p and q are hyperparameters guiding the direction of the random walk. More precisely, if p is large a global exploration of the graph $\mathbf{G}$ is encouraged while, on the contrary, larger values of q bias the walk to a local exploration thus increasing the probability of visiting already seen nodes. Through the use of such parameters, the random walks generated by Node2vec model are a combination of *breadth-first search* (BFS) and *depth-first-search* (DFS) [89].

2. *SkipGram model training*. In the second part of the algorithm the generated random walks are used to train a SkipGram model [174] aimed at obtaining the embedding vectors $z(v_i) \in \mathbb{R}^d$ for every $v_i \in \mathbf{V}$.
SkipGram learns a language model, which maximizes the probability of sequences of words that exist in a training corpus. The objective function used in a SkipGram model for node representation is:

$$\max_z \sum_{v_i \in \mathbf{V}} \log(p(\mathcal{N}(v_i)|z(v_i))), \tag{13}$$

where $\mathcal{N}(v_i)$ is the neighborhood of node v_i sampled from the random walks.
Optimization of Equation 13 involves two distinct assumptions in order to make it tractable:

- We assume that the likelihood of observing a neighbor is independent of observing any other neighbor given the representation of the central node, *i.e.*:

$$p(\mathcal{N}(v_i)|z(v_i)) = \prod_{v_j \in \mathcal{N}(v_i)} p(v_j|z(v_i)). \tag{14}$$

- We further assume that a central node and its neighbors have a symmetric effect on each other in the latent space $\mathbb{R}^d$. Accordingly, we model each term in the product defined in Equation 14 as a softmax function parametrized by a dot product of their representations:

$$p(v_j|z(v_i)) = \frac{\exp z(v_j) \cdot \exp(z(v_i))}{\sum_{v \in \mathbf{V}} \exp z(v) \cdot \exp(z(v_i))}. \tag{15}$$

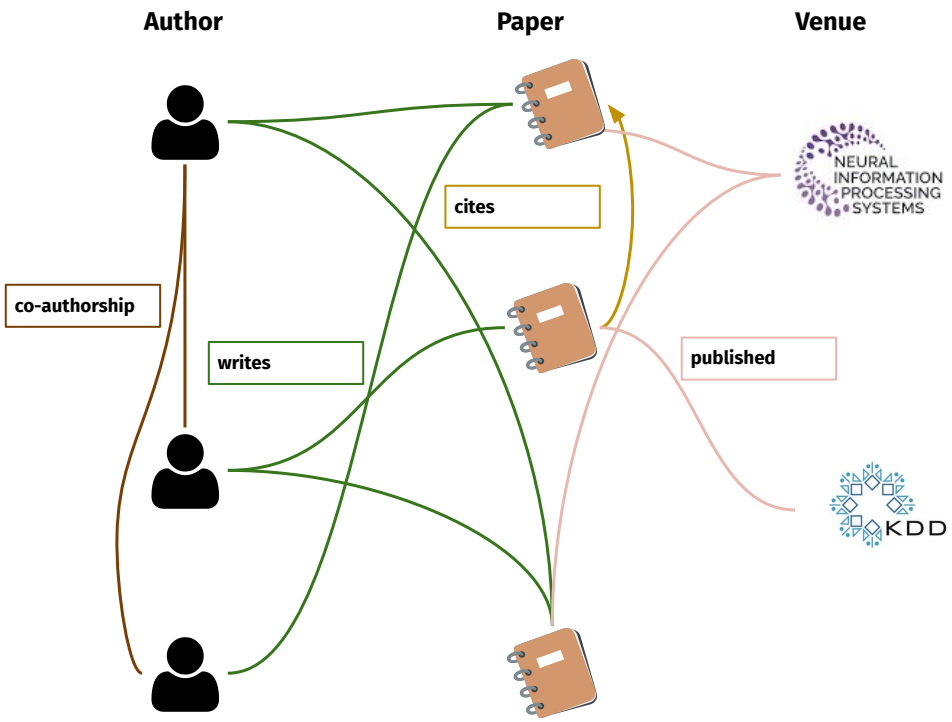


Figure 9: Heterogeneous graph representation of an academic network. Nodes identify authors, papers and venues. Relations among nodes correspond to: co-authorship (researcher-researcher), writes (researcher-paper), cites (paper-paper), published (paper-venue).

Since the dot product between two vectors measures their similarity, by maximizing the softmax function for neighboring nodes, the inferred node representations for neighboring nodes tend to be similar. Computing the normalization constant at the denominator of the conditional probability in Equation 15 is time consuming since it involves the computation of a large number of dot products, especially if the network $\mathbf{G}$ is very large. For such reason, DeepWalk and Node2vec approximate it using hierarchical softmax [201] and negative sampling [108],respectively.

The above mentioned random walk based methods for nodes representation learning are specifically designed for homogeneous networks, meaning networks equipped with a single type of nodes and edges. This is due to the fact that the random walks generated by DeepWalk and Node2ved models do not sample nodes based on either node or edge types. However, it is often the case that the network under analysis is heterogeneous [238], *i.e.* a network equipped with multiple types of nodes and edges connecting them. A typical example of heterogeneous network is represented by academic networks (see Figure 9), which are systems able to describe academic collaborations at different levels of complexity. Representation of such networks poses the problem of preserving node semantics and proximities together in the learnt latent space. In the next section we formally introduce heterogeneous informations

networks (HIN), extend the notion of random walks to metapaths and describe a metapath based method for node representation learning in HINs.

3.2.2 *HINs and metapath2vec*

A *Heterogeneous Information Network* (HIN) [238] is defined as a network $\mathbf{G}$=($\mathbf{V}$,$\mathbf{E}$,$\mathbf{T}$) in which each node $v \in \mathbf{V}$ and each edge $e \in \mathbf{E}$ are associated with their *type functions* $\mu(v) : \mathbf{V} \longrightarrow \mathbf{T_V}$ and $\lambda(e) : \mathbf{E} \longrightarrow \mathbf{T_E}$ respectively. $\mathbf{T_V}$ and $\mathbf{T_E}$ denote the sets of object and relation types, where $|\mathbf{T_V}| + |\mathbf{T_E}| > 2$.

As depicted in Figure 9, we can represent an academic collaboration network with authors (A), papers (P) and venues (V) as nodes, wherein edges indicate the co-authorship (A–A), write (A–P), cite (P-P), publish (P-V) relationships. Within this framework, we can formalize the problem of HIN representation learning as follows.
Given a heterogeneous information network $\mathbf{G}$, we want to learn a latent low-dimensional representation $\mathbf{Z} \in \mathbb{R}^{|\mathbf{V}| \times d}$, with $d \ll |\mathbf{V}|$, able to preserve nodes proximity and relationship semantics.
To such extent, it is of paramount importance being able to suitably define nodes proximity in a HIN and thus nodes heterogeneous neighborhood.
To model the heterogeneous neighborhood of a node, *metapath2vec* [84] introduces the heterogeneous SkipGram model. Such a model differs from the classical SkipGram model (Section 3.2.1) by introducing metapath based random walks in heterogeneous networks to incorporate the heterogeneous network structures into the model. More precisely, a metapath scheme $\mathcal{P}$ is defined as a path represented by $V_1 \xrightarrow{R_1} V_2 \xrightarrow{R_2} \dots \xrightarrow{R_{l-1}} V_l$, where $R = R_1 \circ R_2 \circ \dots \circ R_{l-1}$ identifies the composite relation between node types V_1 and V_l.
For instance, in Figure 9, a metapath "APPA" represents two authors co-cited respective papers. We can leverage metapath to define type-specific random walks. Indeed, given a HIN $\mathbf{G}$, and a metapath scheme $\mathcal{P} : V_1 \xrightarrow{R_1} V_2 \xrightarrow{R_2} \dots \xrightarrow{R_{l-1}} V_l$, the transition probability at step i is defined as:

$$p(v_{i+1}|v_i^t, \mathcal{P}) = \begin{cases} \frac{1}{|\mathcal{N}_{t+1}(v_i^t)|}, & \text{if } (v_{i+1}, v_i^t) \in \mathbf{E},\ \mu(v_{i+1}) = t+1 \\ 0, & \text{if } (v_{i+1}, v_i^t) \in \mathbf{E},\ \mu(v_{i+1}) \neq t+1 \\ 0, & \text{if } (v_{i+1}, v_i^t) \notin \mathbf{E} \end{cases}, \tag{16}$$

where $v_i^t \in V_t$ and $\mathcal{N}_{t+1}(v_i^t)$ denotes the V_{t+1}-type neighborhood of node v_i^t.
Frequencies computed through metapath based random walks can be then used to infer a heterogeneous SkipGram model by the optimization of the following loss function

$$\max_z \sum_{v \in \mathbf{V}} \sum_{t \in \mathbf{T_V}} \sum_{n_t \in \mathcal{N}_t(v)} \log(p(n_t|v, z)), \tag{17}$$

which is analogous to the DeepWalk and Node2vec loss function apart from the sum over node types.

Such representation is demonstrated to capture both the semantic and structural correlations between different types of nodes [239].

3.3 Graph Neural Networks

The previous node embedding approaches we discussed in Section 3.2 use a shallow embedding approach to generate representations of nodes, *i.e.* we simply optimize a unique embedding vector for each node. In this section, we turn our focus to more complex models, namely *graph neural networks*, which are general models aimed at extending the deep learning framework to graph data. Basically, we want to infer node representations that capture both the structure of the graph, as well as any supplementary feature information on nodes.
One simple approach to define a deep neural network over graphs would be to consider a sequence of non linear transformations of the adjacency matrix. The issue with this approach is that it shows a strong dependency on the ordering of nodes in the adjacency matrix $\mathbf{A}$. Equivalently, such a model is not permutation invariant, which is a substantial premise to design neural networks over graphs. More formally, any function f that takes an adjacency matrix $\mathbf{A}$ as input should satisfy one of the two following conditions:

1. $f(\mathbf{S}_n\mathbf{A}\mathbf{S}_n^T) = f(\mathbf{A})$. (permutation invariance);
2. $f(\mathbf{S}_n\mathbf{A}\mathbf{S}_n^T) = \mathbf{S}_n f(\mathbf{A})$ (permutation equivariance),

for every permutation matrix $\mathbf{S}_n$. Ensuring invariance or equivariance is a key challenge when we learn over graphs and it is one of the main reasons that justify the introduction of graph neural networks (GNN).

The basic GNN model can be motivated as a form of neural message passing in which vector messages are exchanged between nodes and updated using neural networks [104]. In the rest of this section, we will describe the neural message passing framework by which, given a graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ along with a matrix of node features $\mathbf{X} \in \mathbb{R}^{p\times|\mathbf{V}|}$, we can generate node embeddings $z_u \in \mathbb{R}^d$ for $u \in \mathbf{V}$.

3.3.1 *Neural Message Passing*

For every node i the graph $u \in \mathbf{V}$, at each iteration k, a hidden embedding $h_u^{(k)}$ is updated according to a function that aggregates messages from the neighborhood of u (see Figure 10). Formally,

$$h_u^{(k+1)} = \text{UPDATE}(h_u^{(k)}, \text{AGGREGATE}(\{h_v^{(k)} \ : \ v \in \mathcal{N}(u)\}) \tag{18}$$

$$= \text{UPDATE}(h_u^{(k)}, \mathbf{m}_{\mathcal{N}_u}^{(k)}), \tag{19}$$

$$\tag{20}$$

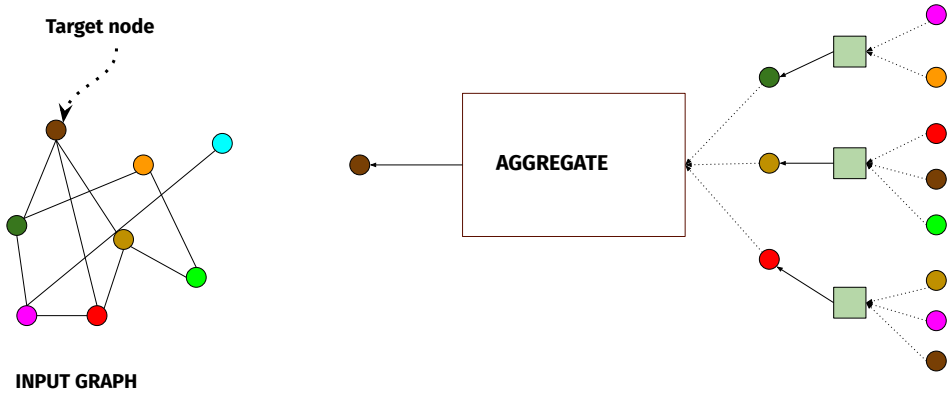


Figure 10: Schematic representation of neighbors messages aggregation by a single node. The model aggregates messages from one node neighbors, and in turn, the messages coming from these neighbors are based on information aggregated from their respective neighborhoods, and so on.

where UPDATE and AGGREGATE are differentiable functions and $\mathbf{m}_{\mathcal{N}_u}^{(k)}$ is the aggregated message diffused to node u.
The AGGREGATE function takes in input the set of embeddings of nodes in $\mathcal{N}(u)$ and generates a message $\mathbf{m}_{\mathcal{N}_u}^{(k)}$, which is then fed to the UPDATE function together with the hidden embedding of node u at the previous iteration, $h_u^{(k)}$, in order to generate the updated embedding $h_u^{(k+1)}$.
At each iteration, every node aggregates information from its neighborhood, and at next iterations each node embedding contains more and more information collected from farther nodes in the graph. More precisely, after the first iteration ($k = 1$), every node embedding contains information from its 1-hop neighborhood, *i.e.*, every node embedding contains information about the features of its first-order proximal nodes; after the second iteration ($k = 2$) every node embedding contains information from its second-order proximal nodes; and in general, after k iterations every node embedding contains information about its k-hop neighborhood.
Each node embedding $h_u^{(k)}$ encodes two types of information:

- structural information: for instance, after k iterations, $h_u^{(k)}$ might collect information about k-hop neighbors degree;

- features information: $h_u^{(k)}$ encodes information about all the features in the k-hop neighborhood of u. in many cases, nodes are equipped with rich sets of features $\mathbf{x}_u \in \mathbb{R}^p$ (*e.g.* textual features in documents corpora, functional features in chemical compounds networks and so and so forth.). However, when node features are not available, several options can be adopted in order to equip nodes with topological features. For example, nodes could be instantiated with vectors of features described in Section 2.2 or with representations obtained by training a random walk-based model (Section 3.2.1). Another popular approach is to use

identity features, where we associate each node with a one-hot indicator feature, which uniquely identifies that node. Nevertheless, this last approach makes the model transductive and incapable of generalizing to unseen nodes.

In order to be able to implement the GNN framework defined in Equation 18, we must define concrete instantiations to the UPDATE and AGGREGATE functions. The most basic GNN framework [222] is defined as

$$h_u^{(k+1)} = \sigma\left(\mathbf{W}_1^{(k+1)} h_u^{(k)} + \mathbf{W}_2^{(k+1)} \sum_{v \in \mathcal{N}(u)} h_v^{(k)} + \mathbf{b}^{(k+1)}\right), \tag{21}$$

where $\mathbf{W}_1^{(k+1)}, \mathbf{W}_2^{(k+1)} \in \mathbb{R}^{d^{(k+1)} \times d^{(k)}}$ are trainable parameters, $\mathbf{b}^{(k+1)} \in \mathbb{R}^{d^{(k+1)}}$ is the bias term and σ is an element-wise non linearity (*e.g.* a ReLU function). Such a model can achieve high performance [115] but generalizations and improvements of both the AGGREGATE and UPDATE functions are available in state-of-the-art.

3.3.2 *AGGREGATE Functions*

The most basic aggregation function is to take the sum of neighboring embeddings as shown in Equation 21. Such an approach may lead to high sensitivity to the presence of hubs in the graph $\mathbf{G}$, leading to numerical instability and problems during optimization [139]. One way to avoid this issue is to simply normalize the aggregation operation by the degrees of the nodes involved

$$\mathbf{m}_{\mathcal{N}(u)}^{(k)} = \frac{\sum_{v \in \mathcal{N}(u)} h_u^{(k)}}{|\mathcal{N}(u)|}. \tag{22}$$

In [139], authors propose another normalization factor, *i.e.* a symmetric normalization

$$\mathbf{m}_{\mathcal{N}(u)}^{(k)} = \sum_{v \in \mathcal{N}(u)} \frac{h_v^{(k)}}{\sqrt{|\mathcal{N}(u)||\mathcal{N}(u)|}}, \tag{23}$$

which they show to be very effective to mitigate the effects of high degree nodes in the graph.

Even if neighborhood normalization can be a useful tool to improve GNN performance, more sophisticated operations have been defined in order to exploit AGGREGATE functions potential. It is worth noting that the neighborhood aggregation operation is fundamentally a set function. This fact is essential since there is no natural ordering of nodes neighbors, and any aggregation function we define must thus be permutation invariant.

Based on the theory of permutation invariant neural networks, in [268] it is shown that any permutation invariant function that maps a set of embeddings to a single embedding can be approximated to an arbitrary accuracy by a model defined as follows:

$$\mathbf{m}_{\mathcal{N}(u)} = \mathbf{MLP}_\phi\left(\sum_{v \in \mathcal{N}(u)} \mathbf{MLP}_\theta(h_v)\right), \tag{24}$$

where $\mathbf{MLP}_\phi, \mathbf{MLP}_\theta$ are multi-layer perceptrons parametrized by ϕ and θ respectively. In Equation 24, we employ a sum of the embeddings after applying the first MLP, however, other aggregate functions, such as element-wise maximum, can be adopted [114, 208].
The model defined in Equation 24, achieves permutation invariance by relying on a sum, mean, or element-wise max to reduce the set of embeddings to a single vector. Nevertheless, other pooling approaches adopt completely different strategies to obtain universal approximators. More specifically, Janossy pooling [183] uses a permutation sensitive function and average the result over many possible permutations. Formally, let $\pi \in \Pi$ be a permutation that maps the set $\{h_v : v \in \mathcal{N}(u)\}$ to the sequence $(h_{\pi(v_1)}, \ldots, h_{\pi(v_{|\mathcal{N}(u)|})})$, the Janossy pooling performs neighborhood aggregation by

$$\mathbf{m}_{\mathcal{N}(u)} = \mathbf{MLP}_\phi\left(\frac{1}{|\Pi|}\sum_{\pi\in\Pi}\rho_\theta(h_{\pi(v_1)}, \ldots, h_{\pi(v_{|\mathcal{N}(u)|})})\right), \tag{25}$$

where Π is a set of permutations and ρ_θ is a parametric permutation sensitive function, *e.g.* a LSTM architecture [130] operating on sequences.
It is clear that if the set Π in Equation 25 is equal to the set of all possible permutations, then the function defined in Equation 25 defines a universal approximator. Nevertheless, considering all possible permutations is usually intractable, thus, in practice, Janossy pooling employs one of two approaches [183]

- random sampling of a subset of possible permutations during each application of the AGGREGATE function and sum over the random subset;
- employ a predefined ordering (*e.g.* according to the degree) of the nodes in the neighborhood set.

NEIGHBORHOOD ATTENTION More general forms of set aggregation can be improved by exploiting *attention mechanisms* [18], closely related to transformer architectures [81, 247, 263] which are mainly used in NLP systems. Basically, we want to assign an attention weight to each neighbor, which is used to weigh its influence during the aggregation step. For instance, the Graph Attention Network (GAT) model [248] uses attention weights to define a weighted sum of the neighbors:

$$\mathbf{m}^{(k)}_{\mathcal{N}(u)} = \sum_{v\in\mathcal{N}(u)} \alpha^{(k)}_{u,v} h^{(k)}_v, \tag{26}$$

where at iteration k, $\alpha^{(k)}_{u,v}$ denotes attention on neighbor v and $h^{(k)}_v$ denotes the neighbor embedding at the k-th layer.
Typical examples of attention weights are defined as

$$\alpha_{u,v} = \frac{\exp\left(\mathbf{a}^T||[\mathbf{W}h_u, \mathbf{W}h_v]\right)}{\sum_{w\in\mathcal{N}(u)}\exp\left(\mathbf{a}^T||[\mathbf{W}h_u, \mathbf{W}h_w]\right)}, \tag{27}$$

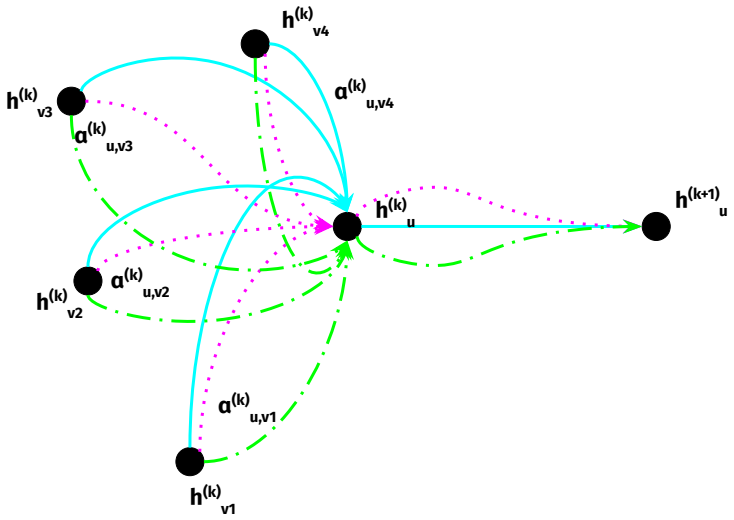


Figure 11: A schematic representation of multi-head attention mechanism (with $H = 3$ heads) by node u on its neighborhood. Different arrow styles and colors denote parallel attention computations. The aggregated features from each head are concatenated or averaged to obtain $h_u^{(k+1)}$.

where $\mathbf{a}$ is a trainable attention vector and $||[\cdot,\cdot]$ represents the common vector concatenation operation. It is also common to include a LeakyReLU non linearity [247] within the attention weight defined in Equation 27, *i.e.*

$$\alpha_{u,v} = \frac{\exp\left(\text{LeakyReLU}\left(\mathbf{a}^T||[\mathbf{W}h_u, \mathbf{W}h_v]\right)\right)}{\sum_{w\in\mathcal{N}(u)} \exp\left(\text{LeakyReLU}\left(\mathbf{a}^T||[\mathbf{W}h_u, \mathbf{W}h_w]\right)\right)}. \tag{28}$$

Further mechanisms, such as *multi-head attention mechanisms* [247], can help stabilize the learning process. More specifically, multi-head attention mechanisms are defined as H independent attention mechanisms that execute the transformation of either Equation 27 or Equation 28, and then their messages are concatenated, resulting in the following output aggregate message (as shown in Figure 11):

$$\mathbf{m}_{\mathcal{N}(u)}^{(k)} = ||_{i=1}^{H} \sum_{v\in\mathcal{N}(u)} \alpha_{u,v}^{k,i} \mathbf{W}^{(k,i)} h_v^{(k)}. \tag{29}$$

3.3.3 *UPDATE Functions*

GNN message passing involves two key steps: aggregation and updating, and the UPDATE operator plays an equally important role in defining the power the GNN model as the AGGREGATE operator. In this section, we turn our attention to more generalizations of the UPDATE operator.

One common issue to most basic GNN models is known as *over-smoothing* [115]. More specifically, it is empirically and theoretically proved that after several iterations basic neural message passing (*e.g.* GraphSAGE [139]), we lose local neighborood information on node representations and the representations for all the nodes in the graph can become very similar to one another. One possible intuition about over-smoothing in GNNs is that in some cases information aggregated (*i.e.* $\mathbf{m}_{\mathcal{N}(u)}$) from node neighbors during message passing begins to dominate the updated node representations (*i.e.* $h_u^{(k)}$). In order to mitigate over-smoothing effects, several improved UPDATE operators have been defined. In particular, vectors concatenation (or *skip connections*) are very useful to preserve information from previous rounds of message passing during the update step. One of the simplest skip-conncections is defined as

$$h_u^{(k+1)} = ||[\text{UPDATE}_{\text{base}}(h_u^{(k)}, \mathbf{m}_{\mathcal{N}(u)}^{(k)}), h_u^{(k)}], \tag{30}$$

where $\text{UPDATE}_{\text{base}}$ is any basic UPDATE operator as the one defined in Equation 21. Here, we attempt to separate the information coming from the neighbors from the current representation of each node when updating its representation.
Other forms of skip-connections have been introduced [205] to alleviate the over-smoothing issue in GNNs, while also improving the numerical stability of optimization.

GNN message passing algorithm can be seen as a system in which an aggregation function receives an observation from the neighbors, which is then used to update the hidden state of each node. This simple observation allows us to apply methods used to update the hidden state of RNN architectures based on observations. For instance, *gated recurrent units* (GRU) [153] can be used to update node representations

$$h_u^{(k+1)} = \text{GRU}(h_u^{(k)}, \mathbf{m}_{\mathcal{N}(u)}^{(k+1)}), \tag{31}$$

where GRU [63] is a gating mechanism similar to a long short-term memory (LSTM) architecture with a forget gate.
In general, any update function defined for RNNs can be employed in the context of GNNs, by simply replacing the hidden state argument of the RNN update function with the node hidden state and the observation vector with the message aggregated from the local neighborhood.

3.3.4 *Heterogeneous GNNs*

Heterogeneous graphs (Section 3.2.2) have been commonly used for modeling complex systems, in which entities of different types interact with each other in several ways. Some instances of such systems include academic graphs [269], Facebook entity graph and LinkedIn economic graph.
One of the classical paradigms to mine heterogeneous graphs is to define and

use metapaths to model heterogeneous structures, such as metapath2vec (Section 3.2.2) [84]. Recently, due to GNNs success, there are several attempts to adopt GNNs to learn with heterogeneous networks [223, 255, 265]. One of the first attempts to generalize homogeneous GNNs framework to heterogeneous networks is to extend classical graph convolutional networks [139] to knowledge graphs. More specifically, given a HIN $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{T})$, we define the following propagation model [223]

$$h_u^{(k+1)} = \sigma\left(\sum_{r \in \mathbf{T_E}} \sum_{v \in \mathcal{N}^r(u)} \frac{1}{c_{u,r}} \mathbf{W}_r^{(k)} h_v^{(k)} + \mathbf{W}_0^{(k)} h_u^{(k)} \right), \tag{32}$$

where $\mathcal{N}^r(u)$ is the relation r-specific neighborhood of node u and $c_{u,r}$ is a problem-specific normalization constant that can either be learned or chosen in advance (*e.g.* $c_{u,r} = |\mathcal{N}^r(u)|$). Intuitively, Equation 32 accumulates transformed feature vectors of neighboring nodes through a normalized sum. Unlike regular GCNs, we introduce relation-specific transformations, *i.e.* depending on the type and direction of an edge.

Even if the model proposed in Equation 32 has demonstrated good performances in several tasks such as link prediction and node classification [223], recent improvements leveraging neighborhood attention (Section 3.3.2) [255] have been proposed. In particular, different metapaths in heterogeneous graphs may extract diverse semantic information and how to select the most meaningful metapaths and fuse the semantic information for the specific task is a hot topic in the research community [152, 226]. Heterogeneous node-level attention [255] is then introduced to learn the importance of metapath based neighbors for each node in a heterogeneous graph and aggregate the representation of these meaningful neighbors to form a node embedding. More formally, given a relation $r \in \mathbf{T_E}$, we define the following relation-specific node-level attention

$$\alpha_{u,v}^{(r,k)} = \frac{\exp\left(\sigma\left(\mathbf{a}_r^T || [h_u, h_v]\right)\right)}{\sum_{w \in \mathcal{N}^r(u)} \exp\left(\sigma\left(\mathbf{a}_r^T || [h_u, h_w]\right)\right)}, \tag{33}$$

where σ denotes a non linearity, and $\mathbf{a}_r$ denotes a trainable vector identifying the node-level attention vector for relation r.

The heterogeneous node-level attention mechanism defined in Equation 33 can be further improved by modelling relation semantics together with node semantics. More specifically, The Relation-aware Heterogeneous Graph Neural Network [265] graph convolution module is meant to propagate information on each relation-specific graph separately and learn node representation specified to the corresponding relation. Moreover, by integrating a cross-relation message passing module, interactions of node representations across different relations are improved. In this framework, the semantic representations of relations are explicitly learned layer by layer to guide the node representation learning process to facilitate downstream tasks and are finally semantically aggregated into a compact representation based on the learned relation representations.

Despite the optimal performances achieved by the aforementioned works, these face several issues:

1. they require the design of customized metapaths for each type of heterogeneous graphs, thus further requiring domain expert efforts;
2. they either simply assume that different types of nodes/edges share the same feature and representation space or learn separate weights for either node type or edge type alone, making them insufficient to capture heterogeneous graphs complexity;
3. they ignore the dynamic nature of heterogeneous graphs.

For this reason, Heterogeneous Graph Transformers (HGT) have been recently introduced [131]. Instead of parameterizing each type of edge, the *heterogeneous mutual attention* in HGT is defined by breaking down each edge $e = (u, v)$ based on its meta relation triplet, *i.e.*, $\langle \mu(u), \lambda(e), \mu(v) \rangle$. As a result, nodes and edges of different types are allowed to maintain their specific representation spaces while letting nodes of different types still interact and aggregate messages without being restricted by their distribution gaps.
Even if HGT does not require to manually design metapaths, the proposed attention mechanism can automatically and implicitly learn and extract metapaths that are important for different downstream tasks. Finally, to handle graph dynamics, the HGT model architecture is equipped with the *relative temporal encoding* (RTE) strategy. Instead of slicing the input graph into different timestamps, all the edges are considered to happen in different times as a whole, and RTE strategy is designed to model structural temporal dependencies with any duration length, and even with unseen and future timestamps. RTE enables HGT to automatically learn the temporal dependency and evolution of heterogeneous graphs by end-to-end training.

3.4 Conclusion

In this chapter we provided background on nodes representation learning methods in homogeneous networks and provided a description of state-of-the-art extensions of such methods to heterogeneous information networks. In particular, we showed how random walk based methods such as DeepWalk and Node2vec leverage network topology to obtain shallow embeddings to generate representations of nodes. Further we saw how deep learning framework can be extended to graph data through graph neural networks. We showed that such architectures, *i.e.* GNNs, provide more complex representations of nodes which may be then used to accomplish inductive learning downstream tasks. In particular, we described recent architectures which implement complex mechanisms, such as *neighborhood attention*, able to characterize local geometry of networks. Parallel to this, we described nodes representation learning in HINs. In particular, we saw how random walk based methods, such as metapath2vec, offer node representations able to capture node semantics

and how recent Heterogeneous Graph Neural Network models overcome the customization of metapaths and learn relation-specific node-level attention weights in order to model complexity in HINs.

4

Contributions

This chapter describes the original contribution of this thesis. In particular, in Section 4.1 we provide an extension of a state-of-the-art method for testing and questioning the overall ubiquitous presence of scale-free networks in real-world complex systems. In Section 4.2 and Section 4.3, we describe applications of classical networks representation learning in two different scenarios, *i.e.* tennis tournaments and preterm infants motion analysis. Finally, in Section 4.4, we propose a preliminary analysis of a heterogeneous information network, *i.e.* the collaboration network of the MaLGa[1] academic research group.

4.1 Statistical Testing of Empirical Power-Law Distributions

Part of this section is present in the following publication: *Garbarino, Davide, Tozzo, Veronica, Vian, Andrea, & Barla, Annalisa (2020, September). A robust method for statistical testing of empirical power-law distributions. In International Workshop on Algorithms and Models for the Web-Graph (pp. 145-157). Springer, Cham.*

The World-Wide-Web is a complex system naturally represented by a directed network of documents (nodes) connected through hyperlinks (edges). In this section, we focus on one of the most relevant topological properties that characterize the network, *i*.e. being scale-free. A directed network is scale-free if its in-degree and out-degree distributions have an approximate and asymptotic power-law behavior. If we consider the Web as a whole, it presents empirical evidence of such property. On the other hand, when we restrict the study of the degree distributions to specific sub-categories of websites, there is no longer strong evidence for it. For this reason, many works questioned the almost universal ubiquity of the scale-free property. Moreover, existing statistical methods to test whether an empirical degree distribution follows a power law suffer from large sample sizes and/or noisy data.

1 https://malga.unige.it/

Here, we propose an extension of a state-of-the-art method that overcomes such problems by applying a Monte Carlo sub-sampling procedure on the graphs. We show on synthetic experiments that even small variations of true power-law distributed data causes the state-of-the-art method to reject the hypothesis, while the proposed method is more sound and stable under such variations.

Lastly, we perform a study on 3 websites showing that indeed, depending on their category, some accept and some refuse the hypothesis of being power-law. We argue that our method could be used to better characterize topological properties deriving from different generative principles: central or peripheral.

4.1.1 *Introduction*

The World-Wide-Web (WWW) encodes associative links among a large amount of pages. Its structure has grown without any central control, thus make it approximable to the result of a random process, where pages link to each other following local probabilistic rules.
Such probabilistic rules are defined through statistical properties of Web graph features. In particular, several investigations show that the WWW is scale-free [8, 22, 50] *i.e.*, both the distributions of incoming and outgoing links are well-approximated by a discrete power law [188]. This can be traced to the fact that the vast majority of documents in the Web have relatively few outgoing and incoming links, but few pages still have enormous number of links that skew the mean of the empirical distribution far above the median.

Nonetheless, when analyzing specific portions of the Web, *i.e.* websites, the scale-free property seems to be less evident especially for specific categories (*e.g.* university homepages) [199, 236]. Note that, differently from what is commonly done in literature [199], we consider websites as closed sub-systems of the Web whose temporal evolution is independent of the system they evolved into.

In this work, we are interested in developing a method able to assess if data from empirical observations follow a power-law. Indeed, testing power laws on empirical data is usually hard due to the large fluctuations that are present in the tail of the distribution.

One of the most commonly used statistical test is the Kolmogorov-Smirnov [67]. This method focuses on the center of the distribution, making it not suitable for testing heavy-tailed distributions. In [67] the authors make strong use of this test by performing a bootstrap procedure that is optimal in small sample size regimes. Indeed, as the sample size grows, the power of the statistical test increases, thus leading to higher rate of rejections of the null hypothesis. Moreover, even in presence of small sample sizes, adding a low amount of noise may cause the test to reject.

As in real-world, noisy or large samples are the common scenario, here, we propose an alternative testing pipeline that leverages on the Anderson Darling test [13] and Monte Carlo sub-sampling. Our pipeline is able to cope with the

power of the test problem by reducing the sample size while maintaining the original degree distribution behavior.

We show synthetic experiments in which the state-of-the-art method fails under small variations or large sample sizes of input data. In all these cases, our method is proved to be more stable under variations and it can be shown that provides results with a better confidence. Lastly, we present case studies on 3 websites representative of different generative processes. These case studies present interesting results showing that indeed, closed sub-portion of the Web do not necessarily follow a power-law distribution. And, they seem to point in the direction that the more the generative process is centralized the less the degree distribution can be associated to a power law decay.

4.1.2 *Discrete power-law distribution: definition, fit and statistical test*

The discrete power-law distribution is defined as

$$p(d_v = x) \approx \frac{1}{\xi(x_{min}, \alpha)} x^{-\alpha}, \tag{34}$$

where d_v is the random variable representing the degree of a node v, x_{min} is a fixed lower bound on the values x, α is a *scaling parameter*, and $\xi(x_{min}, \alpha) = \sum_{x=x_{min}}^{\infty} x^{-\alpha}$ is the Hurwitz-zeta function [117].

The parameter x_{min} is particularly important, as often the degree distribution of a network follows a power law only for degrees x greater than a lower bound. A network is said to be scale-free if the tail of its in-degree and out-degree distributions obeys to a discrete power law decay. In practice, this entails that we have a non-null probability to observe nodes with a degree much greater than average (hubs).

4.1.3 *Maximum Likelihood Estimation*

The parameters x_{min} and α of an empirical power-law distribution need to be estimated from data. Given as input a vector $\mathbf{x} \in \mathbb{N}^n$ representing the degrees of n nodes of a graph, we need to perform two different procedures to estimate these two parameters, as described by the pseudo-code in Algorithm 1.

ESTIMATE OF x_{min} First, we pick $\hat{x}$ as the value that minimizes the difference between the empirical degree distribution and the fitted power-law model where $x_{min} = \hat{x}$ [67, 68].

In order to minimize such difference, we need to select a suitable distance. One of the most common is the Kolmogorov-Smirnov (KS) statistic, which is defined as the supremum norm of the difference between two distribution functions (CDFs) of the empirical data and the best-fit model [166]. Although the KS statistic is widely used, it presents some drawbacks in the detection of

Algorithm 1 Power-law fitting

Input: `degrees` vector of length n
`distances` $= [\,]$
for $x \in \{\min(\texttt{degrees}), \ldots, \max(\texttt{degrees})\}$ **do**
  if len(`unique(degrees)` $> x$) <25 **then**
    break
  end if
  $\alpha \leftarrow$ power_law_fit(`degrees`, $x_{min} = x$)
  $d \leftarrow$ Anderson-Darling(`degrees`, x, α)
  `distances`.append(d)
end for
$\hat{x} \leftarrow \underset{x}{\operatorname{argmin}}$ `distances`
$\hat{\alpha} \leftarrow$ power_law_fit(`degrees`, $x_{min} = \hat{x}$)
$\hat{d} \leftarrow$ Anderson-Darling(`degrees`, $\hat{x}, \hat{\alpha}$)
return $\hat{x}, \hat{\alpha}, \hat{d}$

heavy-tailed distributions since, being based on the CDF, it mainly penalizes fluctuations in the center of the empirical distribution. A more reliable distance for the comparison of heavy-tailed distributions is the Anderson-Darling (AD) statistic as it puts more importance to the extreme values of the CDFs [13]. For this reason, we will recur to this statistic in the rest of the paper. The AD distance is defined as

$$A^2(\mathbf{x}, F_{x_{min}=x}) = -n - \sum_{i=1}^{n} \frac{2i-1}{n} \left[\ln F_{x_{min}=x}(x_i) + \ln(1 - F_{x_{min}=x}(x_{n+1-i})) \right],$$

where n is the sample size and $F_{x_{min}=x}$ is the power-law CDF.

Note that, if we select a $\hat{x} > x_{min}$, we are reducing the size of our training data, and our model will suffer from the statistical fluctuations in the tail of the empirical distribution. On the other hand, if $\hat{x} < x_{min}$, the maximum likelihood estimate of the scaling parameter $\hat{\alpha}$ may be severely biased.

ESTIMATE OF α Given the lower bound x_{min}, we estimate the scaling parameter α by means of maximum likelihood, which provides consistent estimates in the limit of large sample sizes [75].
In the discrete case, a good approximation of the true scaling parameter can be reached mostly in the $x_{min} \geq 6$ regime [67]. And it can be computed as:

$$\hat{\alpha} \approx 1 + n \left[\sum_{i=1}^{n} \ln \frac{x_i}{x_{min} - \frac{1}{2}} \right]^{-1}.$$

4.1.3.1 *Goodness-of-fit test*

Once $\hat{\alpha}$ and $\hat{x}$ have been estimated, we need to assess if observed data are plausibly sampled from the related power-law distribution. To such extent, we perform a goodness-of-fit (GoF) test procedure [169].

Algorithm 2 Power-law testing

Input: `degrees` vector of length $n, \hat{x}, \hat{\alpha}, \hat{d}$
`distances` $= [\,]$
for $i = 1, \ldots, M$ **do**
  $n_{tail} = \text{count}(\texttt{degrees} > \hat{x})$
  for $j = 1, \ldots, n$ **do**:
    $b \leftarrow \text{bernoulli\_sample}(n_{tail}/n)$
    if b is 1 **then**
      $s_i[k] = \text{power\_law\_sample}(\hat{x}, \hat{\alpha})$
    else
      $s_i[j] \leftarrow \text{uniform\_sample}(\texttt{degrees} < \hat{x})$
    end if
  end for
  $\alpha_i, x_i \leftarrow \text{power\_law\_fit}(s)$
  $d \leftarrow \text{Anderson-Darling}(s, x_i, \alpha_i)$
  `distances`.append(d)
end for
`p-value` $= \text{count}(\texttt{distances} > \hat{d})/M$
return `p-value`

A goodness-of-fit test measures how well a statistical model fits into a set of observations. Given the statistical model under testing, a GoF makes use of a statistic that evaluates the discrepancy between the observed values and the expected value of the model. By definition, a statistic is a function which does not depend on the parameters of the model. The output of the GoF procedure is a p-value corresponding to the probability that the statistic is greater than its realization on the observed data.

Note that, since we estimate the model parameters from data we do not know the distribution of the statistic. Thus, we perform a semi-parametric bootstrap approach to estimate such distribution empirically[67, 237].

In particular, we fixed as statistic the Anderson-Darling distance and we perform a procedure described in Algorithm 2. Given n samples, we indicate with n_{tail} the amount of samples that are greater than $\hat{x}$. Bootstrap is then performed by simulating n_{tail} examples from a power law with parameters $\hat{\alpha}$ and $\hat{x}$, and for the remaining sample size $n - n_{tail}$ we sample degrees from the empirical data that are smaller than $\hat{x}$. We repeat this procedure M times. The value of M depends on the desired significance of the p-value. Typically, if we want a p-value that approximates its true value with an error smaller than ϵ, then $M = \frac{1}{4\epsilon^2}$.

Given the M simulated data sets, we fit to each of them its own power-law model and compute the AD distance. This provides the empirical distribution of the AD statistic that we use to compute the associated p-value, defined as the fraction of synthetic distances larger than the observed one.

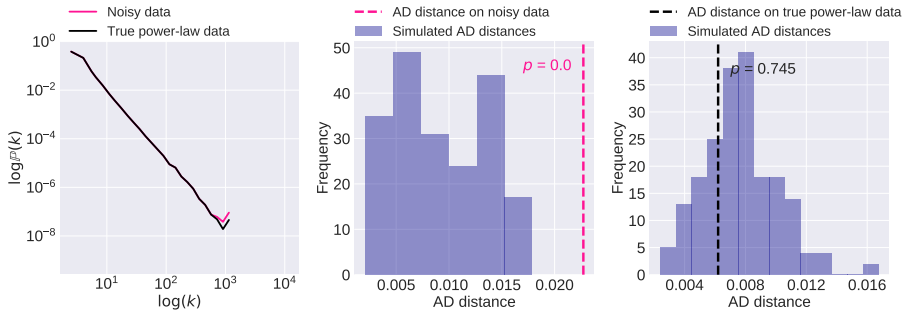


Figure 12: On the left, the empirical probability density functions of true power-law data (black line) and noisy power-law data (pink). On the right, the Anderson-Darling test on both samples. Little variations from an exact power-law sample lead to reject the null hypothesis.

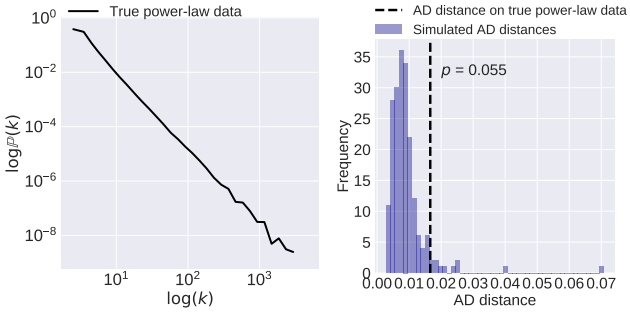


Figure 13: On the left, the empirical probability density function of true power law data. On the right, the Anderson-Darling test. Large sample size (5×10^5) leads to reject the null hypothesis.

If p is large (relatively to a fixed significance level, *e.g.* 0.1), we cannot reject the null hypothesis. Then, possibly, the difference between the empirical and theoretical distributions may be attributed to statistical fluctuations. Differently, if p is smaller than the significance level, we say that the empirical data are not power law.

4.1.4 *Problems of goodness-of-fit on empirical data*

Testing whether empirical data are power-law distributed is a hard task. This is due to the following reasons: a) the probability of rejecting the null hypothesis grows with sample size; and, as a consequence b) the procedure is too sensitive to even minimal amount of noise. Little attention has been put on these issues, but we argue that they are crucial as they heavily affect the final response of the statistical test.

In particular, both problems can be addressed by considering the *power of the test*, which, fixed a significance level, is defined as the probability of correctly rejecting the null hypothesis. Such probability increases accordingly to the sample size, hence, when the number of nodes n is large, we tend to reject the null hypothesis even in cases of true power-law distributed data (as the power

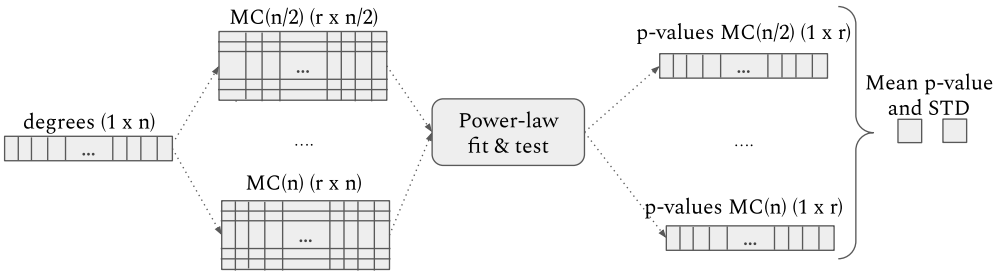


Figure 14: Schematic representation of the proposed pipeline.

of the test is very close to 1). Indeed, by performing bootstrap, we simulate nearly exact power-law samples, which induce the Anderson-Darling test to be very sensitive to even minimal fluctuations in the observed distribution.

In Figure 12 and 13, we show two synthetic experiments where such test fails, in particular:

1. we generated $n = 10^5$ samples from a discrete power-law distribution with parameters $x_{min} = 7$ and $\alpha = 2.7$. We perturbed the data by adding one occurrence to the last 13 degrees in the extreme tail (see Figure 12 left panel for the true and perturbed data);

2. we generated $n = 5 \times 10^5$ samples from a discrete power-law distribution with parameters $x_{min} = 2$ and $\alpha = 2.7$.

We applied the procedure in Section 2 on both datasets, with $M = 200$ and significance level set to 0.1. Results are shown on the right side of Figure 12 and 13. In Figure 12, the empirical probability density functions of the two samples are indistinguishable from each other except in the extreme tail, where little divergences can be traced. Thus, it becomes evident that for large sample sizes the test is very sensitive even to little fluctuations in the observed sample. Also, with example (b) we show that even perfect power-law samples induce the test to fail when the sample size is too large (Figure 13).

Both examples show that the high power of the Anderson-Darling test in large sample size regimes constitutes a drawback of the previously introduced method [67]. Since it is never the case that an observed degree distribution is exactly drawn from a discrete power law, we propose a variation of the method in Section 2 that aims at testing the goodness of fit of heavy tail distributions.

4.1.5 *Monte Carlo approach*

Our proposal is based on the idea of performing iterative Monte Carlo (MC) sub-samplings of different length on the original degree sequence. We argue that with this sub-sampling scheme we can reduce the sample size without modifying the trend of the original degree distribution and possibly obtain a more reliable test.

The global scheme of the procedure is provided in Figure 14. In particular, we define a set of lengths, $\{l_1, \ldots, l_{max}\}$, for each length we perform r corresponding MC samplings. For each sample, we fit a power-law distribution

and assess its plausibility exploiting Algorithm 1 and Algorithm 2and, thus, obtaining a sequence of p-values of the Anderson-Darling test of length r. We consider, as final output of the procedure, the mean of all p-values sequences for all different lengths and the related standard deviation.

To the best of our knowledge, it is not usual to exploit MC sub-sampling to test for power-law decay in the degree distribution. In fact, performing MC does not allow to exactly estimate the parameters of the power-law distribution, indeed, to each sub-sample may correspond a different set of parameters. Nonetheless, we do not use MC as a fitting method but rather to say if a network is plausible to asymptotically satisfying the scale-free property. We argue that using MC as a way to obtain suitable sub-samples of smaller sample size would provide better understanding of the degree sequence behavior while overcoming the drawbacks induced by large sample sizes.

4.1.5.1 *Instantiation of parameters*

In order to apply the Monte Carlo approach we need to fix different values, specifically l_1, l_{max}, r and the significance level.

The problem of selecting adequate lengths for the MC sub-samples is not trivial. On the one hand, a too small sub-sample would lead to very different degree sequences due to the large fluctuations present in the original network, while, on the other hand, lengths close to the original degree sequence would lead to higher rates of rejection of the power-law hypothesis. Then, we arbitrarily decided to set l_1 at $n/2$ which is half the length of the observed data. As for l_{max}, we fix it to n as in case of true power-law samples we want to being able to obtain a high p-value, while in case of noisy data, considering one length equal to the original size does not particularly affect the resulting mean p-value.

The value of r affects the robustness of the final result, the more repetitions the better approximation of the true p-value. Nonetheless, its value depends on constraints deriving from computational power. Thus, we leave the definition of such value to the user.

We fixed the significance level at 0.1 for the rejection of the null hypothesis. This is a conservative choice implying that the power law hypothesis is ruled out if there is a probability of 1 in 10 or less that data sampled from the true model agree with the model as the empirical data.

Lastly, we fixed the maximal possible x_{min} to be least 25 observations less than the maximal observed degree. This is due to limit the chances of fitting a power-law distribution on too few observations.

4.1.6 *Experimental results*

In order to evaluate the performance of the proposed pipeline, we perform four experiments and compare the results with the state-of-the-art method. In the rest of the narration we will refer to the state-of-the-art method as Bootstrap and to our method as Monte Carlo + Bootstrap.

Test type	Erdős-Renyi		Barabasi-Albert		
	75000	150000	75000	150000	300000
Bootstrap	0.00 ± 0.00	0.00 ± 0.00	0.85 ± 0.24	0.75 ± 0.27	0.78 ± 0.18
MC+Boostrap	0.00 ± 0.00	0.00 ± 0.00	0.85 ± 0.06	0.69 ± 0.16	0.71 ± 0.15

Table 1: Results to assess the goodness of the proposed testing pipeline in cases of scale-free graphs (Barabasi-Albert) or not (Erdős-Renyi), in terms of mean p-value and standard deviation on 10 repetitions of the test for different sample sizes.

All the simulations are performed in Python. We used the package `powerlaw` [12] for fitting power-law distributions to empirical data and compute the AD distances. We provide all the notebooks used for the experiments of this paper in a GitHub repository[2]. For all experiments, we fixed 30 lengths of Monte Carlo re-sampling in the interval $[\frac{n}{2}, n]$ and for each of this length we get $r = 10$ re-samplings.

4.1.6.1 *Validation of the proposed method on different graph models*

In the first experiment we aim at verifying if Monte Carlo + Bootstrap is comparable to just Bootstrap when considering two cases at varying sample sizes:

1. Erdős-Renyi models of size $\{75 \times 10^3, 15 \times 10^4\}$, we expect both methods to refuse the null hypothesis as the degree distribution of this model is known to follow a binomial distribution [88]. Thus, we use this as base test to assess the probability of correctly rejecting the power-law hypothesis.

2. Barabasi-Albert models of size $\{75 \times 10^3, 15 \times 10^4, 3 \times 10^5\}$, we expect both methods to have high p-values as the degree distribution follows a power law [25]. We use this experiment to provide proof of the soundness of the method in presence of true power-law data.

Each experiment listed above is repeated 10 times to estimate the mean and standard deviation of p-values. Results are reported in Table 1 where we observe that our approach (Monte Carlo + Bootstrap) always reject the null hypothesis in the Erdős-Renyi case as the Bootstrap method, while in the Barabasi-Albert case we always provide p-values with a smaller variance.

4.1.6.2 *Robustness to noise*

We now want to assess that our method is indeed more robust under increasing noise in the input empirical distribution. We simulated from a discrete power law with parameters $\alpha = 2.3$ and $x_{min} = 1$, a sample of size $n = 10^5$. For

2 https://github.com/DaviGarba/netanalytics

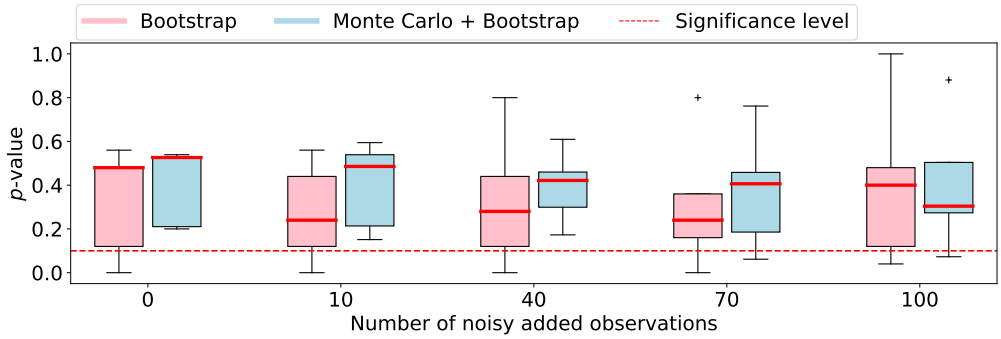


Figure 15: Results in terms of p-values for the two testing pipelines as the input data present an increasing level of noisy observations.

different levels of noise in the set $\bar{n} \in \{10, 40, 70, 100\}$, we perturbed the power law observation by adding $\bar{n}$ values uniformly sampled from the original observation.

Figure 15 shows that the proposed methods is in mean always better than the simple bootstrap approach while also providing a smaller variance. Also, it never reject the null-hypothesis in cases in which the noise is small while sometimes it rejects it in presence of high amount of noise (100 added observations). Differently from the Bootstrap approach that, depending on the simulated sample, sometimes rejects it even in presence of zero noise.

4.1.6.3 *Benchmark on University of Notre Dame website*

We exploit a widely studied example of empirical data that is assumed to follow a power-law distribution [8, 21, 181], *i*.e. the web graph of the University of Notre Dame website. This graph, in 1999, has been studied in order to obtain information regarding the topology of the Web. In [8], the authors found that the in-degree and out-degree distributions of the graph underlying the hyperlink structure of the domain nd.edu were well approximated by power-law distributions with scaling parameters 2.7 and 2.1 respectively. We downloaded the hyperlink graph from http://snap.stanford.edu/ [149]; the crawl consists of 325729 documents and 1497134 links. We tested the Monte Carlo + Bootstrap approach against the Bootstrap approach as the empirical data are noisy and we want to provide further validation of our testing procedure on the in-degree distribution of the network.

We performed $r = 5$ MC re-samplings for different sizes equally spaced in the interval $[162864, 325729]$. Monte Carlo + Bootstrap results in a mean p-value of 0.15, meaning that there is no strong evidence against the power-law hypothesis for the in-degree distribution. Differently, when applying the Bootstrap method we observed a p-value equal to 0.00, which would lead us to reject the null hypothesis.

As in literature many have argued the power-law nature of this graph, this allows us to conclude that our testing procedure is more robust and thus can be applied on real-world data with higher reliability.

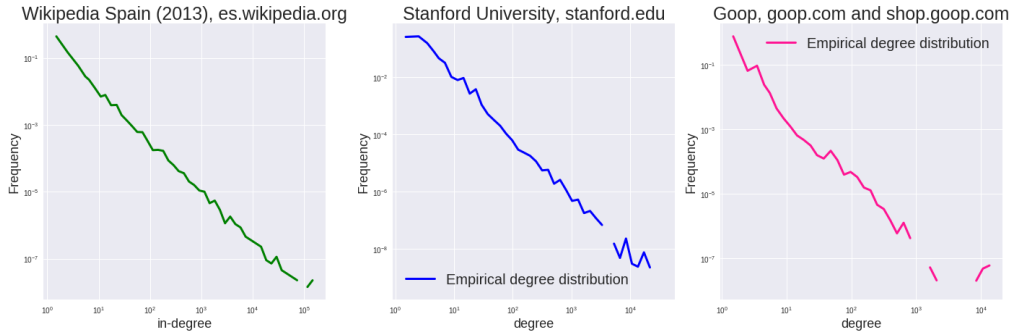


Figure 16: Log-log plots of the empirical distributions of the considered case studies.

4.1.6.4 *Website analysis*

We now want to exploit our procedure in real scenarios to seek for evidence of differences in the degree distributions deriving from different generative processes. We considered three different websites that we deemed representative of different strategies of content creation: e-commerce, academic and free encyclopedia. The first category is typically characterized by a strong central control in the design and evolution of the information architecture and content generation. Conversely, the last category is completely user-guided and its evolution is, thus, likely to be random. We argue that the academic category, as well as other website of complex institutions, should be a trade-off between the two, as usually many contributors have access to writing and adding content with a mild central control.

We consider the following websites:

1. Goop, the website of a wellness and lifestyle company; we crawled the entire website using the open source framework Scrapy[3], during the crawl we restricted to the domains `goop.com` and `shop.goop.com`;
2. Stanford, the website of Stanford University. We downloaded a crawl performed in 2002 available at http://snap.stanford.edu/;
3. Wikipedia (ES), the website of the free spanish encyclopedia. We downloaded a crawl of 2013 at http://law.di.unimi.it/index.php [42, 43].

Table 2 describes the characteristics of the three considered websites, in terms of category, number of nodes and number of edges. Table 2 also reports the mean p-values obtained with Monte Carlo + Bootstrap on the in-degree distributions. Results seems to validate our hypothesis about an inverse correlation between the centrality of the content generative process and the scale-free property.

4.1.7 *Discussion*

In this section, we proposed a method for hypothesis testing of power-law distributions in empirical data that overcomes issues related to the power of

3 https://scrapy.org/

Name	**Url**	**Website type**	**No. Nodes**	**No. Edges**	p-value
Goop	goop.com	E-commerce	100.482	731.259	0.00 ± 0.00
Stanford	stanford.edu	Academic	281.903	2.312.497	0.01 ± 0.00
Wikipedia(ES)	es.wikipedia.org	Encyclopedia	972.933	23.041.488	0.74 ± 0.21

Table 2: Analyzed websites with the related information about number of nodes, number of edges and category.

the test. In particular, our method mediates the effect of possibly noisy data through Monte-Carlo sub-samplings of the empirical distribution. We verified that the proposed method retains the ability of assessing if observations are indeed plausibly sampled from a power law, under different sample sizes and level of noise. Indeed, the method is more reliable than the state-of-the art on synthetic data. To further assess the reliability of our approach we also provide a real-world example, specifically the University of Notre Dame website, which is a well studied dataset and it is considered to be scale-free. Our method does indeed provide a p-value higher than the significance level, differently from the state-of-the-art method that rejects the null hypothesis.

This allowed us to use our method to test different websites corresponding to different content generative processes. From our first insights, we observed that different content generation strategies may induce a different connectivity structure of the hyperlink graph.

For future research we intend to increase the number of real networks studied and consider current websites related to different generative processes to provide a more comprehensive understanding of specific sub-categories of the Web.

Future research directions may also involve the use of random walks instead of Monte Carlo as a sub-sampling technique on graphs [27, 159] and the comparison with other estimators of power laws in empirical data [214].

To conclude, our pipeline is an attempt to perform statistical testing while considering its limits both theoretical and due to noisiness of data. We argue that this is fundamental to reliably test assumptions on real-world examples.

4.2 Predicting Tennis Match Outcomes with Network Analysis and ML

Part of this section is present in the following publication: *Bayram, Firas, Garbarino, Davide, & Barla, Annalisa (2021, January). Predicting Tennis Match Outcomes with Network Analysis and Machine Learning. In International Conference on Current Trends in Theory and Practice of Informatics (pp. 505-518). Springer, Cham.*

Singles tennis is one of the most popular individual sports in the world. Many researchers have embarked on a wide range of approaches to model a tennis match, using probabilistic modeling, or applying machine learning models to predict the outcome of matches. In this section, we propose a novel approach based on network analysis to infer a surface-specific and time-varying score for professional tennis players and use it in addition to players' statistics of previous matches to represent tennis match data. Using the resulting features, we apply advanced machine learning paradigms such as Multi-Output Regression and Learning Using Privileged Information, and compare the results with standard machine learning approaches. The models are trained and tested on more than 83,000 men's singles tennis matches between the years 1991 and 2020. Evaluating the results shows the proposed methods provide more accurate predictions of tennis match outcome than classical approaches and outperform the existing methods in the literature and the current state-of-the-art models in tennis.

4.2.1 *Introduction*

4.2.1.1 *Motivation*

The sports industry is one of the fastest growing business sectors in the world. According to the Business Research Company, the global sports market reached a value of nearly $488 billion in 2018, having grown at an annual growth rate of more than 4% since 2014, and is expected to reach almost $614 billion by 2022. As we live in the age of data and analytics, this steady growth rate for the sports market size has motivated many researchers to conduct studies on sports data analytics where in sport competitions a result can convey a great deal on different aspects involved in sports like the volume of fans retention, television contracts or sponsorship deals. Essentially, sports data analytics is exploited by either the sports teams directly or by sports gambling stocks. One primary technique of predictive analytics is to build machine learning models to generate predictions for upcoming events and matches using historical player data and statistics.

Several leading male tennis professionals like Roger Federer, Novak Djokovic and Andy Murray have realized the importance of data analytics in tennis, so they introduced data analytics specialists into their teams to help them better prepare for tournaments. They scrutinize and analyze their opponents' key

skills and tactics to make use of those insights to avail themselves of the opportunity to boost their chances of winning matches. Accurate prediction of the outcome of tennis matches has an impact on advising players of their odds so they can adjust their plans according to the forecast of the match.

4.2.1.2 *Related Work*

Complex network techniques have been applied to represent the network of tennis matches [83]. The majority of approaches were to rank the players in tennis history taking into account a global view of the player's performance throughout his career and compare it to the existing system that ATP is currently following, which is to rank tennis players based on the immediate past 52 weeks. The most notable work on tennis network modeling was done by Radicchi [209] where the author determined the best players on specific playing surface and proposed a ranking algorithm *Prestige Score*, that is analogous to PageRank score [48], to quantify the importance of tennis players and concluded that the prestige score is more accurate and has higher predictive power than ranking schemes adopted in professional tennis. Michieli [173] applied multiple ranking algorithms to see how active tennis players have improved their overall prestige over the recent years and compared the results of the ranking methods used with the ATP Ranking and identified *Jimmy Connors* as the best player in history up to 2017. Breznik [47] identified the best left and right-handed players in tennis history by applying network analytic methods and the PageRank algorithm.

For tennis match prediction, most existing approaches to tennis prediction apply statistical models to tennis matches, starting from the hierarchical structure of the sport's scoring system under the assumption that points in tennis are independently and identically distributed or i.i.d, Klaassen and Magnus [140] show that the assumption is false but they find that deviations from i.i.d. are small and hence the i.i.d. assumption provides a reasonable approximation. O'Malley [192] and, Klaassen and Magnus [141], are illustrative examples of such models. Knottenbelt et al [142]improved the hierarchical model that is based on the probability of winning an individual point by exploiting statistics from matches played against common opponents. In recent years, machine learning models have been utilized to predict the winner of a tennis match by representing the match and the player by a set of features instead of a single value, player's features are derived from historical match statistics. Ma et al[162] applied logistic regression model on 16 variables representing player skills and performance, player characteristics and match characteristics. Sipko and Knottenbelt [229] have extracted more detailed set of features and applied logistic regression and artificial neural network to predict the outcome of a tennis match, their best model ANN resulted in a log loss of 0.6111. Peters and Murray [202] addressed the effects of surface type and the variation of player skills through time by using free parameters to represent the skills of players and the characteristics of court surfaces.

Here, we define a new method based on network analysis to extract a new feature that represent the player's skill on each surface considering the varia-

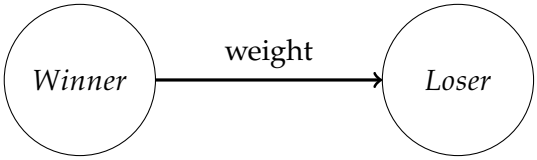


Figure 17: A single tennis match represented in a directed graph representation

tion of his performance over time which is believed to have a big effect on the match outcome. We also make use of the match statistics directly in the prediction through applying advanced ML paradigms instead of only following the historical averaging process that was done in the literature. This project uses data obtained from the ATP official website which is the main resource for historical data of tennis matches since 1968. Each match is represented by 49 features including, for instance, player's age and ranking, the number of aces, double faults and 1^{st} serve percentage.

4.2.2 *Network Modeling and Surface-Specific Score*

In this section, we describe the method followed to extract the surface-specific score.

4.2.2.1 *Network of Tennis Matches*

We mapped tennis matches into a weighted and directed graph: edges are directed from winner to loser and they are weighted according to the stage and type of tournament as shown in Fig. 17. The ATP has four tiers of events– Grand Slams, Masters 1000, ATP 500 and ATP 250. With the four Grand Slams awarding the most points, 2000. The numbers in each tournament category represent how many points the winner receives. For clarity, we simplified the numbers that we used to assign weights to the edges to hold the proportion between the values as follows: ATP 250/500 : 1, Masters 1000: 2, ATP Finals: 3, Grand Slams: 4. In case of multiple links of the same direction exist between the players, the weights are summed up.

4.2.2.2 *Surface-specific score*

It is evident in tennis that players' performances are affected by the court surface, Barnet and Pollard [26] showed that the type of surfaces favors those players who are best suited to this particular surface. The dataset in hand does not include any information about players' skills on each surface. Therefore, in order to quantify the latter, we subset the graph based on surface and all prior tennis matches, we compute several centrality measures for each player and, by means of PCA, we evaluate a surface-specific score. Centrality measures are a widely used analysis mechanism to reveal important elements of complex networks [76]. Note that all the used measures are normalized.

CM1 **Out-In-degree-difference Centrality:** The in-degree of a player v (d_v^{in}) is the sum of the edge weights of his losing matches, while out-degree (d_v^{out}) is the sum of the edge weights of his winning matches. The Out-In-degree-difference centrality measure (CM1) of each node is computed as the difference between its in-degree and out-degree.

CM2 **Hubs Centrality:** The hub score of a node estimates how many highly authoritative nodes this node is pointing to; in the tennis players network, Michieli [173] demonstrated that good hubs are often associated with successful players because they have won against a wide range of players while the authorities are modest players with long careers.

CM3 **PageRank Centrality:** PageRank centrality [194] is a spectral centrality measure. The algorithm assigns a centrality score based on its neighbors score. PageRank acknowledges that not all wins are equal, wins over strong opponents weigh more than beating mediocre players. PageRank score of player i is computed as following:

$$P_i = \frac{d}{N} + (1-d)\sum_j P_j\left(\frac{w_{ji}}{k_j} + \frac{\delta(k_j)}{N}\right) \tag{35}$$

Where $d = 0.15$ is the damping factor, N is the number of players, w_{ji} is the edge weight between nodes i and j, k_j and P_j are the out-degree and PageRank value of the node j, respectively and δ is a function to correct the sinks (nodes with outdegree zero).

Finally, to estimate the surface-specific score, we followed Algorithm 3, based on the Principal Component Analysis (PCA) technique.

Algorithm 3 Surface-Specific Score Extraction Algorithm

```
Input: Graph of matches on specific surface G^(s), Centrality measure functions C
Output: Vector of surface-specific scores y^(s)
for For each node i in graph y^(s) do
    for For each function c in C do
        CM(c) = C(i)
        y^(s)(i) = PCA(CM)
    end for
end for
return y^(s)
```

We refer to the resulting first principal component scores as *surface-score*. These values can be interpreted as time-varying surface-specific scores. Table. 3 and Table. 4 show the top 5 scores on Clay and Hard surfaces respectively, as of June 2020.

4.2.3 *Tennis Match Representation*

In this section, we discuss how we processed the raw data to generate the features that, in addition to the surface-specific score, are used as input to the ML models.

4.2.3.1 *Player's Features*

There are two types of features in our dataset with respect to their availability time. Unlike some features which are available before the start of the match, such as the age and rank of the players, the statistics are only available after the end of the match. Therefore, player's skills are estimated by taking the average of the statistics of his historical matches for an upcoming match.

4.2.3.2 *Labelling - Experimental Design*

The raw data classifies the players as a winner and loser, whereas before the match takes place, we only have players labelled as Player 1 and Player 2. Thus, we randomly sampled the data and assigned Player 1 to be a *winner* or *loser*. The match outcome for match n can be defined as following:

$$y_n = \begin{cases} 1, & \text{if Player 1 is the winner.} \\ 0, & \text{if Player 1 is the loser.} \end{cases} \tag{36}$$

4.2.3.3 *Symmetric Representation*

Inspired by Sipko and Knottenbelt [229], and O'Malley [38], to achieve the symmetry of model's prediction outcome regardless of the random labeling discussed in the previous section.We took the difference between the players' features of the same characteristic in order to obtain identical results even if we swap the labelling of Player 1 and Player 2. Also, in this way, even supposing that we reverse the labels of Player 1 and Player 2 we are going to get the same feature values but with different sign, and different target class. This would also help us avoid any bias to the same feature of both players due to assigning different weight of a feature for Player 1 than Player 2 by the model. For example, the model might give a higher weight for Player 1 rank than to

Player	Surface-score
Rafael Nadal	0.539565
Novak Djokovic	0.262964
Roger Federer	0.252947
David Ferrer	0.206311
Guillermo Vilas	0.150491

Table 3: Top 5 scores on Clay, as of June 2020

Player	Surface-score
Roger Federer	0.627906
Novak Djokovic	0.566555
Rafael Nadal	0.375329
Andy Murray	0.320519
Andre Agassi	0.23004

Table 4: Top 5 scores on Hard, as of June 2020

Player 2 rank. Using the difference of variables to extract the features halves the number of dimensions of the dataset. The difference of the variables is calculated based on the labeling criterion defined in Equ. 36 as follows:

$$FEATURE_i = STAT_{i,p1} - STAT_{i,p2} \tag{37}$$

4.2.4 *Machine Learning Methods*

We think of a tennis match as a vector x_i composed of P input features. The corresponding match outcome y_i may be a *win* 1 or a *loss* 0.

As for the supervised classification methods, we resorted to four methods. Here we briefly describe the approaches and how we used them to predict the tennis match outcome.

RF **Random Forests** Random Forests is an ensemble technique that combines bagging and random feature sub-spacing. Random Forests algorithm constructs a large collection of classification or regression ensembles of independent decision trees (forests) and aggregates their predictions by averaging [38].

LR **Logistic Regression** Logistic regression exploits the logistic sigmoid function as loss function to estimate the probability of the sample being assigned to one of the two possible classes [121].

LUPI **Learning Using Privileged Information** Learning Using Privileged Information, or Support Vector Machine using Privileged Information (SVM+), has been first proposed by Vapnik and Vashist [246]. LUPI is an advanced learning paradigm that uses additional (privileged) information that is available only for the training examples and not available for test examples. This additional information (prior knowledge) can be exploited to build better models and improve the results. In tennis, and sports in general, the match statistics are only available in the training examples as they are collected during the match and the final stats sheet is published after the end of the match. On the other hand, for the testing examples, we only have the match characteristics and players' profiles, thus we utilized LUPI paradigm to leverage the match statistics that are considered to be as additional information we have for the training examples to improve the performance of the learning method. In LUPI framework, we are given the training triplets:

$$\{(x_1, x_1^*, y_1), \ldots, (x_n, x_n^*, y_n)\},\ x_i \in X,\ x_i^* \in X^*,\ y_i \in \{0,1\},\ i = 1, 2, \ldots, n$$

where X is the space of match features vector x, x^* represents the match statistics vector that belongs to the correcting space X^* (the space of privileged information) and y_i is the labels vector. Supporting the learning process by incorporating the match statistics in the model can capture the complexity of the training examples by discovering hidden patterns between the players that cannot be discovered in the original features space

X. There are many examples in the tennis world where Player 1 has better estimates than Player 2- this is going to be reflected by positive match features and SVM might map the match data point x to the *winning* side of the margin in space X. But still, if Player 1 struggles against Player 2 under specific conditions or due to his play style, this is going to be evident in the statistics of the matches between the two players, hence the additional information x^* that belongs to the same point x might fall in the *losing* side of the margin in space X^*. Therefore, the match statistics can facilitate the learning process by tightening or relaxing the SVM constraints to improve the predictions by including the privileged information that we have about the training matches.

MTR **Multi-Target Regression** Multi-target regression, also known as multi-output regression [44], is an advanced machine learning paradigm that involves predicting simultaneously two or more numerical output variables given the same input features. We utilized MTR approach to predict the statistics of the players in the match and then consequently predict the winner of the match applying classification model. Fig. 18 shows the workflow diagram of implementing the MTR paradigm to predict the winner of the tennis match. The workflow diagram shows that the MTR phase works as a 'black box' to the match outcome prediction since it will only help the classification model better predict the winner of the match and the MTR prediction accuracy in itself is irrelevant to the final evaluation of our learning classification task. The most common approach to deal with multi-target regression problems is *problem transformation* by transforming the multi-target problem into multiple independent single-target problems each one solved by fitting a regressor to make a single-output regression for each target. The other approach is *algorithm adaptation* methods that modify a single-output method to simultaneously support multi-output problems, this is usually done by modeling the dependencies among these targets.

We briefly describe four different approaches to MTR that we adopted in our classification pipeline:

MTSR **Multi Single-Target Regressors:** we decompose the problem to d single-target regression problems by fitting a random forest regressor to independently predict each target [44].

MTR-RC **Regressor Chains:** RC [232] is a *problem transfer* method and is built on the idea of linking chains of regressors and stacking predictions to other models of the chain as additional features. The training procedure of RC involves selecting a chain that is represented by an ordered set of target variables $C = \{y_1, y_2 \dots y_d\}$. To predict the first target value y_1, we trained a random forest regressor on the original input vector X of the training dataset. Then, for the subsequent targets y_j where $j \in \{2, \dots, d\}$ we perform transformation on the training dataset to consist of the union of the original input vector

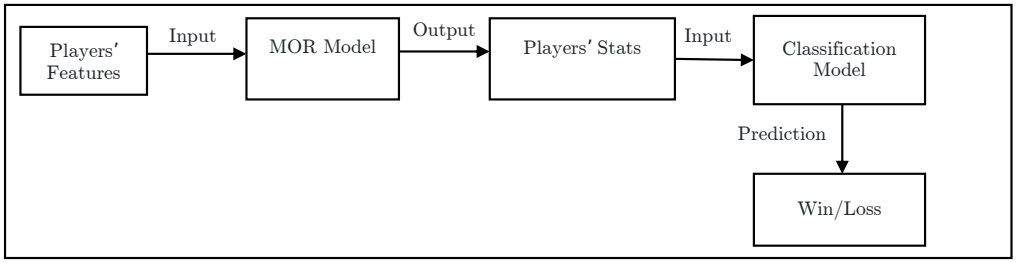


Figure 18: Multi-output Regression Workflow Diagram

X and the actual value of the previous targets in the chain $y_{k<j}$ in the original training dataset.

MTR-RF **Multi-Target Random Forest Regressor:** this method [46, 150] is an extension of the single-target random forest regressor; the only difference is the modification of the calculation of the impurity measure of a node as the sum of squared error over the multi-target value.

MTR-TSF **Multi-Target Regression Via Target Specific Features:** this method, proposed by Wang et al [253], deals with the multi-target regression (MTR) tasks by learning target specific features (TSF). The method assigns a cluster index to each match features vector X_i using hierarchical clustering algorithm. The index is then added to expand the feature space $X_{\text{exp}} = X \cup X_{\text{index}}$. Target specific features are learned by querying a corresponding dependent similarity matrix, generated by a classification and regression tree boosting method (CART-boosting)[58]. The transformed training dataset for the jth match statistics target Y_j is constructed by finding the union $\widehat{D}_J = \mathrm{X} \cup \mathrm{X}_{\text{index}} \cup \mathrm{X}_{\text{TSF}}$.

4.2.5 *Experiments*

The experiments were conducted on a node hosted on DLTM[4], the node is equipped with an Intel Xeon CPU with 27x2.3 GHz, 64 GB RAM.

4.2.5.1 *Dataset splitting*

For the standard ML models, the training dataset consists of 62,141 tennis matches between the years 1991 and 2011, and 21,083 matches between 2012 and 2020 for testing, that makes up approximately 75:25 ratio. For the advanced ML paradigms and because of their complexity, we reduced the dataset size to consist of 39,033 tennis matches between the years 2001 and 2014 for training dataset, and 13,011 matches between 2015 and 2020 for testing, that also makes up approximately 75:25 ratio. For each dataset, we used the tennis matches played in the last three years of the training set for validation.

4 https://www.dltm.it/

This way of dividing the tennis dataset by complete years is widely adopted in the literature. The main reason is to maintain the temporal order of tennis matches. Shuffling the data randomly would result in testing the model on older matches than the ones used for training, which is not legitimate. For example, it would be futile to predict the winner of a tennis match played in 2006 using machine learning model trained on data that include tennis matches played in 2016. Moreover, ATP has a fixed calendar of certain tournaments played in specific weeks of the season. Splitting the matches by complete years allows us to have match samples of each tournament over the different dataset splits. In other words, we have match examples of each tournament distributed in the training, validation and test sets, which makes the learning process more feasible.

4.2.5.2 *Classical Machine Learning Models Results*

Before fitting our model to make the final predictions, we perform feature selection step using sequential backward selection method to select the best features of the dataset. Since the approach requires setting the number of features to be selected a priori, we tuned this parameter and evaluated the performance of the selected subset of the features using the validation set. For RF model, the optimal number of features is 6, while for LR model, the feature selection approach resulted in no improvement in the accuracy while evaluating on the validation set. Table. 5 compares the results of RF and LR models when evaluated on the testing set using accuracy and log loss classification as metrics. Tehe Random Forests algorithm outperformed logistic regression when implemented with feature selection step and without. When comparing the prediction results between the two algorithms using various classification metrics, random forest with feature selection resulted in higher prediction accuracy, and also improved the uncertainty of the predictions measured by the log-loss.

Model	Log-Loss	Accuracy
RF without FS	0.6095	0.6681
RF with FS	0.5996*	0.6729*
LR	0.6110	0.6663

Table 5: Random forests with and without feature selection (RF with FS and RF without FS respectively) and logistic regression results in terms of Log-Loss and Accuracy.

To inspect the impact of the surface-score feature on the prediction results, we used the Shapley Value [161] as shown in Fig. 19. To calculate Shapley Values of each feature in the set, a model is trained with that feature present, and another model is trained with the feature withheld. Then, predictions from the two models are compared on the current input. We see that the **RANK** vari-

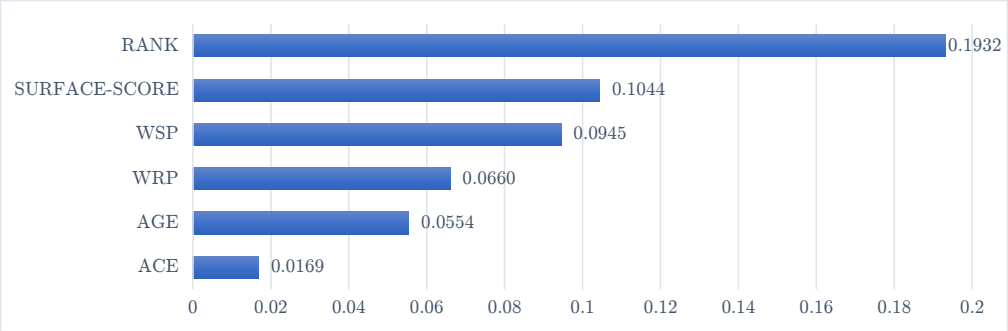


Figure 19: Average impacts (in absolute terms) of features on model output magnitude, WSP is winning on serve percentage, WRP is winning on return percentage

Table 6: Accuracy results of SVM+ with different kernels. SVM, random forest and logistic regression are applied on the original features

Decision Space	Correcting Space Kernel			Original Features		
Kernel	Linear	RBF	Sigmoid	SVM	RF-OF	LR-OF
RBF	0.66159	0.6612	0.6622*	0.655	0.657	0.654
Linear	0.64822	0.64927	0.6561	0.64768		
Sigmoid	0.6526	0.6503	0.6607	0.6305		

able is the most important feature in making the predictions, then **SURFACE-SCORE** feature that we inferred based on network analysis.

4.2.5.3 *SVM and SVM+ Results*

Table. 6 provides a summary of the accuracy results of applying SVM+ using different kernel function combinations in decision and correcting spaces. We also report the accuracy results of the classical SVM algorithm using the corresponding kernel functions. Since we have used different years range as discussed in Section 4.2.5.1, and for the sake of comparing, we applied the classical machine learning models on the same dataset. The standard ML models, RF and LR, are trained on the original features of the dataset without using the match statistics, named *RF-OF* and *LR-OF* respectively. We can see the improvement in the models' performance when leveraging the match statistics as privileged information to correct the decision function. Also, considering a RBF kernel has provided the highest accuracy percentage using both SVM and SVM+ models. Furthermore, the models in LUPI framework have superior predicting performance when compared to RF and LR. SVM+ improved the classification accuracy results of both models.

4.2.5.4 *Multi-Output Regression Results*

To predict the statistics of the match, we fitted several multi-output regression models and used the mean squared error metric to evaluate the performance of

each model. Table. 7 reports the predictive results for both regression and classification phases for the different models. We can notice the big impact of the output of the regression phase on the final classification predictions. Improving the mean squared error of the MTR models by 10^{-3} leads to obtain higher accuracy results by 1.5%. Comparing the performance of the different MTR approaches, we can see that the models which consider the dependency between the features have better results than the ones that neglect it. This supports the hypothesis that tennis match statistics are correlated and modeling this relationship between the features is a powerful technique that should be followed to achieve higher prediction accuracy. Specifically, multi-target regression via specific target features (MTR-TSF) has had the best regression performance. Consequently as a result, the classifier that was built on the MTR-TSF predictions as an input has the highest accuracy results compared with the other MTR models.

Table 7: Regression and classification results of multi-target regression via specific target features (MTR-TSF), multi single-target regression (MTSR), multi-target regression via regressor chain (MTR-RC), Multi-Target Random Forest Regressor(MTR-RF), random forest on the original features (RF-OF), and Logistic regression on the original feature (LR-OF)

	Regression Phase **Mean Squared Error for Each Target**						**Classification Phase**
Approach	M-ACES	M-DFS	M-W1S	M-W2S	M-WSP	M-WSG	Accuracy
MTR-TSF	26.63	8.29	0.0191	0.0281	0.0162	0.0610	0.666*
MTSR	27.60	8.71	0.0204	0.0297	0.0173	0.0647	0.65
MTR-RC	26.39	8.34	0.020	0.0295	0.0173	0.0665	0.65
MTR-RF	27.27	8.54	0.198	0.0291	0.0168	0.0631	0.649
RF-OF	-						0.657
LR-OF	-						0.654

4.2.6 *Conclusions*

4.2.6.1 *Contribution*

In this section, we utilized network analysis techniques and applied advanced machine learning paradigms to improve the current state-of-the-art approaches to predict the winner of a tennis match. We developed a novel method by representing the tennis matches as a network to infer a time-varying and surface-specific score that evaluate the player's performance on a specific court surface at a certain time point. We made use of the extracted score to enhance our dataset and added it to the features set that contains estimations of players' qualities based on the historical matches. We demonstrated that the extracted score has a relevant influence on the prediction results and was ranked as the second most important feature in making predictions of our classification task.

We made use of advanced machine learning paradigms (LUPI and MTR) which resulted in more accurate results when compared to the classical machine learning models. By using these advanced methods, we improved the prediction accuracy results of the classification task by 1.5%. We do emphasize that the proposed methods can outperform the current state-of-the-art models, and can be even generalized to other sports that have a similar data structure, i.e., where the match statistics are only available for training and not available for testing. The advanced paradigms leverage the match statistics directly in making predictions instead of solely using statistics estimators (for example historical average) as per usual when applying the classical machine learning models.

4.2.6.2 *Limitations*

Implementing the advanced machine learning paradigm is associated with additional cost and complexity. This cost in both the resources and the execution time slows down the optimization and hyperparameters tuning process to select the best model that produces the optimal performance and results. Building the kernel matrices for LUPI, and generating the dependent similarity matrix for MTR-TSF, are the most time-consuming and costly phases of the models.

4.2.6.3 *Future Work*

The limitations of the approaches described in the previous section prompt us to think about models that are memory and run-time efficient. Online learning would be a natural candidate. In online learning paradigm, the model is quickly updated to produce the best model as the data arrive in a sequential order. Thus, there is no need to re-train the model whenever a new data point arrives which is too expensive. Online learning is an option worth exploring in predicting tennis match outcomes as it has a rich literature.

Our dataset only includes the totals of winning points on serve and return. In fact, there is a plenty of other aspects in the game of tennis that differentiate between the players' qualities and skills. Acquiring more statistics, such as winners and unforced errors records, head-to-head results on a specific surface, or success rate in winning points at the net, can improve the models' performance. It would also be useful to model the non-numerical factors and convert them into numbers to include them in the dataset, such as the player's current form or favoring specific events which can play a part in helping predict the winner of the match.

4.3 Attributed Graphettes-based Preterm Infants Motion Analysis

Part of this section is present in the following publication: *Garbarino, Davide, Moro, Matteo, Tacchino, Chiara, Moretti, Paolo, Casadio, Maura, Odone, Francesca, & Barla, Annalisa (2021, November). Attributed Graphettes-Based Preterm Infants Motion Analysis. In International Conference on Complex Networks and Their Applications (pp. 82-93). Springer, Cham.*

The study of preterm infants neuro-motor status can be performed by analyzing infants' spontaneous movements. Nowadays, available automatic methods for assessing infants motion patterns are still limited. We present a novel pipeline for the characterization of infants' spontaneous movements, which given RGB videos leverages on network analysis and NLP. First, we describe a body configuration for each frame considering landmark points on infants bodies as nodes of a network and connecting them depending on their proximity. Each configuration can be described by means of *attributed graphettes*. We identify each attributed graphette by a string, thus allowing to study videos as texts, *i.e.* sequences of strings. This allows us exploiting NLP methods as topic modelling to obtain interpretable representations. We analyze topics to describe both global and local differences in infants with normal and abnormal motion patters. We find encouraging correspondences between our results and evaluations performed by expert physicians.

4.3.1 *Introduction*

The analysis of preterm infants motion is a crucial and complex task. The World Health Organization (WHO) [258] has highlighted that among preterm infants (*i.e.*, infants born before 37 completed weeks of gestation), there is a 5-15% chance of developing motor alterations caused by permanent lesions of the developing brain [30] that commonly involve areas of the brain intended for control of movements. An early diagnosis of abnormal motion patterns would allow the start of early rehabilitation treatments, increasing the chances of recovery. In this direction, due to the increase of preterm survival rate in high-income countries [11], a lot of effort has been made to find an automatic and reliable way to characterize and analyze preterm infants neuro-motor status based on the characterization of infants' spontaneous movements [127].

An accurate quantitative analysis of human motion is usually performed with wearable sensors, markers and motion capture systems [70, 170]. Unfortunately, markers and sensors placed on the body skin are cumbersome and they can affect the naturalness of the motion [52], especially in infants [127]. For these reasons, recently, marker-less techniques for human motion analysis based on computer vision have been studied [70, 80] and applied to the analysis of infants motion [3, 6, 53, 101]. These techniques have the potential to solve or reduce some of the issues of marker-based approaches, as they allow for a

more natural person-friendly interaction, they are non invasive and not expensive. Furthermore, they can be adopted to characterize motor configurations not easily detectable with markers.

We approach the problem of representing spontaneous movements sequences of preterm infants by studying it as a temporal network analysis problem. More precisely, we map each frame of a video to a 5-node graph whose nodes are landmark points and edges are inserted based on the distance of the landmark points on the image plane. As far as we know, this is an original approach, never used before.

We model the networks as sequences of 5-nodes *attributed graphettes* [118], defined as not necessarily connected, non-isomorphic induced subgraphs of a larger graph, whose nodes are equipped with attributes.

We want to exploit this modelling choice in order to obtain an interpretable, low-dimensional representation of each video, able to convey information about the local dynamics of each infant. In this sense, in [156] authors present a work in which they define a representation of a large social network by using methods of topic modelling [9]. Specifically, In [156] authors build topic models (that they call *structural topics*) upon graphettes occurrences in node neighborhoods by using anonymous walks to approximate their concentrations in the network. Such topic models, in which graphettes are actually encoded as words of a text that is the network, allow to overcome the only description of networks through graphettes concentrations [207] by including the distribution on graphettes themselves. This allows us to obtain a representation of the network as a mixture of structural topics which in turn are outlined as a mixture of graphettes. In our problem, we leverage on the method described in [156], instantiated with a Latent Dirichlet Allocation [40] model, to identify local motion patterns able to characterize infants' spontaneous movements. This method allow us to highlight insightful differences between the classes of infants with normal (N) and abnormal (Ab) motion patterns. In particular, the motion of infants in Ab class is better characterized by highly symmetric configurations and lower variability. On the contrary, infants in N class are characterized by less symmetric configurations and higher motion variability.

4.3.2 *Materials and Methods*

4.3.2.1 *Dataset*

Data acquisition was performed 3 months after an infant's birth and involved 118 preterm infants (one video for each infant, 78 females, born at 29 ± 2 weeks and weighting 1150 ± 303g). The acquisition setup was composed by a single RGB camera (Canon Legria HF R37, acquiring at 25 frames per second with a resolution of 1080x1920 pixels) placed on a support above a treatment table. We excluded from the analysis those portions of videos where interventions of the operators occluded part of the scene or where the infants were crying.

Among the 118 infants included in this study, 53 had a clinical diagnosis of neuro-motor disorders, presenting a wide spectrum of motor disorders inten-

sities but only a minority with major impairments. The neuro-motor assessments performed 30 months after the video recording were based on different clinical evaluations and tests, including the Bayley test [31] for the majority of the infants and Magnetic Resonance Imaging (MRI) at birth. The study and the consent form signed by parents were approved by the Giannina Gaslini Hospital Institutional Review Board on 20/06/2013 (protocol number: IGGPM01).

4.3.2.2 *Pre-processing: landmark points detection and filtering*

In our setting, we use as nodes of the networks representation a set of meaningful landmark points detected on infant bodies. To this aim, we rely on a deep model for semantic features detection, DeepLabCut (DLC) [167], suitably trained to detect nose (**N**), left hand (**LH**), right hand (**RH**), left foot (**LF**) and right foot (**RF**). From our dataset, we randomly select 10 frames from 100 videos and we manually label the points of interest. Among the different deep architectures provided in DLC, we select a ResNet-50 [123] pretrained on Imagenet [79] and with a final deconvolutional layer to extract spatial density probability maps associated with each landmark point. The architecture is trained with the parameters suggested in [167]. We adopt our trained model to extract the positions of the five landmark points in all the frames for each infant's video, as shown in Figure 20. For each video, the outputs of the model are $\{(x_l^t, y_l^t, c_l^t)\}_{t=1}^T$, with $l = \{\mathbf{N}, \mathbf{LH}, \mathbf{RH}, \mathbf{LF}, \mathbf{RF}\}$. The $l-$th point in the $t-$th frame is identified by its position (x_l^t, y_l^t) and likelihood c_l^t, a number in the interval $[0, 1]$. The coordinates obtained are then filtered in order to improve the

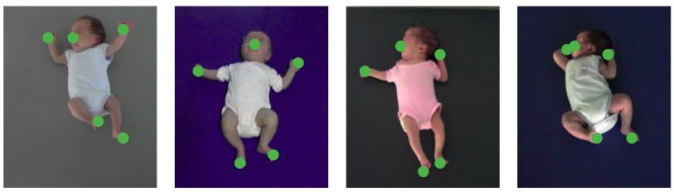


Figure 20: Examples of detected landmarks in the image plane. Images cropped for visualization purpose.

stability across time of the estimated points and discard mispredictions. Focusing on c_l^t, we are able to quantify the uncertainty behind the detection of each point in each frame. We consider as wrongly detected points with $c_l^t < 0.75$. Then, in order to recognize other possible mispredictions, we drop points corresponding to high peaks in the speed profile of each coordinate. We discard points with these characteristics and, if the information loss lasts less then 2 seconds (50 frames), we interpolate the trajectories in order to reconstruct the information. Finally we smooth the resulting signals with a low-pass filter (Butterworth, 4th order, 10 Hz cut-off frequency).

4.3.2.3 *Networks definition*

For each video, we build a temporal sequence of networks (one per frame) describing the relation among the landmark points of interest in the image plane,

used as nodes, that are connected through edges depending on their relative proximity. More specifically, edges are obtained by computing the Euclidean distance between every pair of landmark points in all the images composing our dataset. For each infant, all the computed distances are normalized by the maximum distance across the whole video in the image plane between the nose and the virtual middle point between the feet. Distances normalization compensates for possible differences both in the size of infants' body and in the distances between the camera and the acquisition plane. We use the normalized distances distribution to identify which points are close to or far from each other at each timepoint.

In order to privilege sparser networks for the ease of analysis, we assume to be unlikely for two landmark points to be often close to each other. Therefore, we state that if the distance between two landmark points is greater than the 25th quartile of the corresponding empirical distribution, then they are far from each other, and we do not connect them with an edge. Conversely, we link two nodes with an edge if their normalized Euclidean distance is lower than the 25th quartile of the corresponding empirical distribution. This assumption allows us to define a binary temporal network for each infant in the dataset: Figure 21 shows the first 10 configurations of an infant's sequence represented by their adjacency matrix. The choice of the threshold is driven by the necessity for sparsity yet arbitrary. Possible alternatives will be further explored in future works.

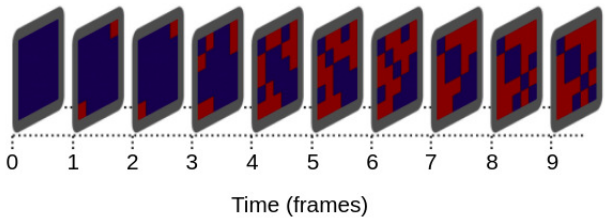


Figure 21: Ten consecutive layers of an infant network represented as a sequence of adjacency matrices.

Each layer in an infant network represents a configuration at a specific timepoint (frame) and it is defined as a 5-node graph, which we exchangeably call *attributed graphette* or *configuration*. In the remainder of the paper, we leverage this representation to describe infants motion in an unsupervised fashion.

4.3.2.4 *Attributed graphettes-based representation*

Each temporal network built for each infant and defined in Section 4.3.2.3 can be represented as a sequence of configurations describing infants motion patterns. More formally, the t-th layer, corresponding to the t-th frame, of a temporal network G is represented by a graph $g_t = (V, E_t, L)$, where $V = \{1, 2, 3, 4, 5\}$ is the set of nodes, E_t is the set of edges and $L = \{\mathbf{N}, \mathbf{RH}, \mathbf{LH}, \mathbf{RF}, \mathbf{LF}\}$ is the set of node attributes. It is important to note that at each timepoint t, the map assigning a node $n \in V$ to a label $l \in L$ is a bijection. Furthermore $|E_t| \in \{0, \dots, 10\}$, thus allowing for the presence of not connected subgraphs

in every infant video. Such subgraphs are called *graphettes* [118], defined as not necessarily connected, non-isomorphic induced subgraphs of a larger graph. Similarly to graphlets [207] and motifs [176], graphettes are a suitable tool to give a local and global description of large complex networks. Indeed, by computing node-level graphettes concentrations in a network we are able to describe local wiring patterns [244] and, at the same time, by aggregating this local information, we get a global description of the network based on the occurrences of these substructures [207]. It is usually a hard task to develop exhaustive graphettes enumeration algorithms, especially when dealing with attributed graphettes, as they are much larger in number than those without attributes [82]. Nevertheless, we exploit the work of [118] and the nature of our problem to define an exhaustive enumeration algorithm.

ATTRIBUTED GRAPHETTES ENUMERATION ALGORITHM Knowing that the number of possible configurations representing one frame is limited to 2^{10}, we generate all the possible 5-nodes attributed graphettes and represent them with the upper triangle of their adjacency matrix unravelled into a bit vector. Each bit vector is then associated to a *word*, *i.e.*, a string composed by the letter g and an integer number ranging from 0 to 1023. A practical example of this process is shown in Figure 22.

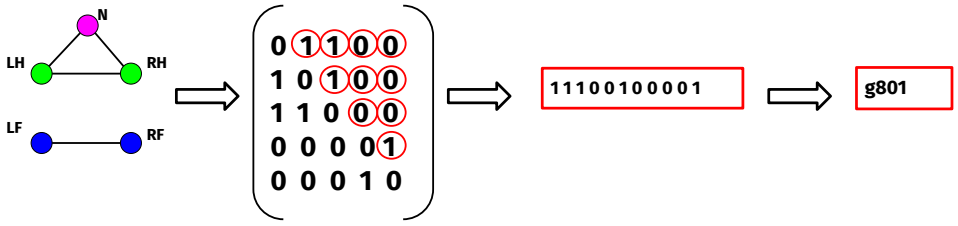


Figure 22: Canonical representation of one instance of a 5-node attributed graphette.

Given an infant video G, we sequentially associate each frame t of the video with an attributed graphette, represented by the corresponding string g_n. For instance, the attributed graphette 0000000000 corresponds to the string g_0. A network is then defined as an ordered sequence of elements from the set $\{g_0, \ldots, g_{1023}\}$, whose length is equal to the number of frames of the corresponding video. Indeed, G results as a collection of configuration names that we treat as *text*, resorting to Natural Language Processing (NLP) methods for text representation in order to enumerate attributed graphettes and describe infants motion in terms of their occurrences.

In this regard, the Bag-Of-Words (BOW) [105] model is a histogram representation that transforms any text into fixed-length vectors by counting how many times each word appears in. This vectorization process is performed by fixing or inferring a *vocabulary*, which is contained in or equal to the set of all words found in the documents. In our case, the vocabulary of all configurations appearing in the dataset consists of 650 attributed graphettes. Therefore, after fitting a BOW model, every infant's network turns out to be a vector of

size 1×650. Figure 23 (left panel) offers a visual representation of a video as a BOW vector.

In order to identify those configurations that are discriminative for networks in the dataset, we need to normalize raw counts in BOW vectors properly. For this purpose, we leverage on Term Frequency - Inverse Document Frequency (tf-idf), a common algorithm to transform word counts into meaningful real numbers [219]. More specifically, given a configuration g_n and a network G, tf-idf measures the originality of g_n by comparing the number of times g_n appears in G (*i.e. term frequency*) to the number of networks g_n appears in (*i.e. document frequency*).

To reduce the dimensionality of these representations, we set a threshold on the minimum and maximum document frequency of configurations. In the tf-idf case, we also retain the ability of weighting graphettes based on their commonality in the dataset. Figure 23 (right panel) illustrates a tf-idf transform (with minimum and maximum document frequencies set to 45% and 70% respectively) of a network in the infants dataset.

Latent Dirichlet Allocation Even if tf-idf approach provides an arbitrary amount of reduction in description length, it does not reveal any information on intra-networks distribution over all attributed graphettes. To overcome this limitation, we resort to topic modeling [9] to define an interpretable low-dimensional representation of videos, able to describe the distribution of attributed graphettes for each infant and also able to give local information on the dynamic of infants by considering co-occurrences of configurations.
Topic models [9] are probabilistic generative models for large collections of textual data (*i.e.*, text corpora). A notable topic model is Latent Dirichlet Allocation (LDA) [40] defined as a 3-level hierarchical Bayesian model, in which every item in a corpus is modelled as a mixture over an underlying set of topics, which are, in turn, described by a probability distribution over words. Topic probabilities offer an explicit low-dimensional representation of texts which has been recently adopted to analyse large social networks [156]. In the remainder of the paper we adopt an LDA representation of infant networks built upon the tf-idf transformations of attributed graphette counts.

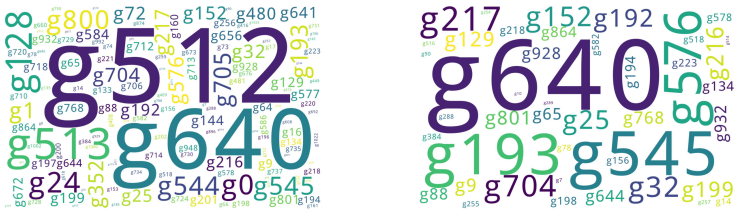


Figure 23: BOW (left) and tf-idf (right) *word cloud* visualization of an infant's temporal network. The size of configuration names is proportional to their weights in the corresponding representation. Note that the configuration $g512$ is either very frequent or rare in the collection of infants networks and therefore it has weight equal to 0 in the tf-idf representation.

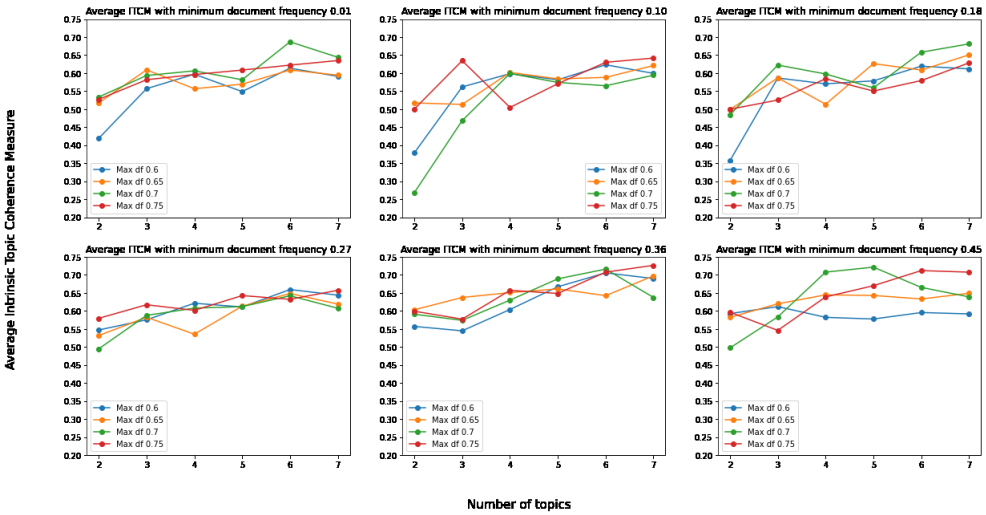


Figure 24: Average ITCM evaluated for number of topics (NoT) ranging in $\{2,3,4,5,6,7\}$ and for different values of maximum and minimum document frequencies in the tf-idf representation. As shown in the bottom right corner, the optimal choice is $NoT = 5$, maximum df equal to 70% and minimum df equal to 45%.

DATA AUGMENTATION Typically, in order to obtain reliable and stable topics, LDA needs to be trained on a large amount of data. Our dataset is composed of 118 infants (65 with normal and 53 with abnormal motion patterns) which is too small to infer meaningful topics from LDA. Then, in order to augment the dataset, we simulate videos from the two classes (*i.e.*, infants with normal (N) and abnormal (Ab) motion patterns) until we obtain a balanced dataset of 1130 videos (118 original videos, 500 and 512 simulated videos from the classes N and Ab, respectively). Simulated networks are composed of 10710 consecutive configurations, which is the average number of frames composing original infants videos. We simulate temporal networks by leveraging normalized bigrams (*i.e.*, couples of adjacent configurations) counts from the original dataset. More specifically, given a temporal network G we compute bigram frequencies and associate every configuration g_n with a vector $v_{g_n} = (bf_i)_{i=1}^{650}$ where bf_i corresponds to the normalized frequency of the bigram $(g_n\, g_{n(i)})$ in G, $g_{n(i)}$ identifying the i-th configuration in the vocabulary. Thus, for every infant, we obtain a matrix X_G (650×650) describing an infant-specific conditional distribution over configurations. We generate networks by first picking at random an infant from a chosen class and a starting configuration, then we iteratively sample configurations from the probability distribution identified by X_G.

NUMBER OF TOPICS SELECTION One of the most crucial LDA hyperparameters that needs to be tuned is the number of topics. In literature, many metrics have been defined in order to find an optimal number of topics [40, 216]. We focus on the maximization of the *Intrinsic Topic Coherence Measure* (ITCM) [177],

which is a metric based on the co-occurrence of words within the documents being modeled. For every topic p, ITCM is defined as

$$ITCM(p, V^p) = \sum_{m=2}^{M} \sum_{h=1}^{m-1} \log \frac{df(v_m^p, v_h^p) + 1}{df(v_h^p)}, \tag{38}$$

where $V^p = (v_1^p, \ldots, v_M^p)$ is a list of the M most probable configurations in the topic p, $df(v_m^p, v_h^p)$ is the number of documents where the configurations pair v_m^p and v_h^p appear together in, and $df(v_h^p)$ is the document frequency of the configuration v_h^p. For each topic, co-occurrence frequencies of the M most probable configurations ($df(v_m^p, v_h^p)$ in Equation (38)) are computed within fixed-size temporal windows for every network. We consider $M = 10$ and a temporal window size equal to 110 frames. We select an optimal number of topics (NoT) by studying how ITCM varies as NoT ranges in $\{2, 3, 4, 5, 6, 7\}$ when applying LDA to the tf-idf transform of the augmented dataset for different settings of maximum and minimum document frequencies. Maximum and minimum document frequencies values are chosen based on the original dataset statistics. More specifically, minimum document frequencies range in $\{1\%, 10\%, 18\%, 27\%, 36\%, 45\%\}$ where 1% corresponds to 1 infant in the original dataset and 45% correspond to the total amount of infants with abnormal motion patterns. Similarly, maximum document frequencies range in $\{60\%, 65\%, 70\%, 75\%\}$ where we set 60% as lower bound as we assume that every configuration appearing in more than half of the infants is non-discriminative. As shown in Figure 24, we obtain that an optimal ITCM is reached at $NoT = 5$, maximum and minimum document frequency equal to 70% and 45%, respectively.

4.3.3 *Results: topics analysis*

By fitting LDA with such hyperparameters to the augmented infants dataset we obtain 5 topics describing local motion patterns as a result of the ITCM maximization. Figure 25 shows the topics summarized by their 5 most probable configurations. Topic-specific most probable configurations differ from each other only by few edges and also appear as little modifications of a basic configuration. This is evident by looking at the first 2 most probable configurations in Figure 25. For instance Topic 2 is well summarized by the configuration in which the only present edges are the ones which connect a hand with the corresponding foot, meaning **LH-LF** and **RH-RF**. Indeed the 2 most probable configurations appear as slight deviations from this basic configuration.

Then, we study topic proportions for every network in the original dataset in order to look for differences between the networks representation of infants with normal and abnormal motion patterns. Topic proportions of networks provide us with a global description of infants movement. Indeed, for each network in the dataset, larger mixture components correspond to topics whose most probable configurations are peculiar to the corresponding infant's motion sequence. Furthermore, topic proportions are suitable to be interpreted as

probabilistic assignments to clusters, which are identified by the corresponding topics.

We perform class-specific topic proportions analysis, as reported in Table 8. In particular, for each network in the dataset, we observe the largest mixture component in its topic representation, that tells us the confidence in assigning the network itself to the corresponding topic. Once assigned the infants to the corresponding prevalent topic, we compute intra-topic, class-specific mean, minimum and maximum probability assignments. We claim that such statistics are good descriptors of the variety of intra-class motion. Also, for each topic, we compute the concentrations of infants in N and Ab classes assigned to it. Differences in such concentrations would indicate different global motion patterns between the two classes. Furthermore, for each topic, we evaluate the mean global symmetry and density of the 5 most probable configurations as well as the mean symmetry of hands and feet neighborhood. In general, from Table 8 we can observe that:

1. no significant differences are detected in the concentrations of infants assigned to each topic.

2. Infants with normal motion patterns are more uniformly distributed among the 5 different topics meaning that they present a higher variability in terms of motion patterns.

3. Infants with abnormal motion patterns are well represented in Topic 0 and Topic 4 (considering the minimum and the mean probability assignments respectively).

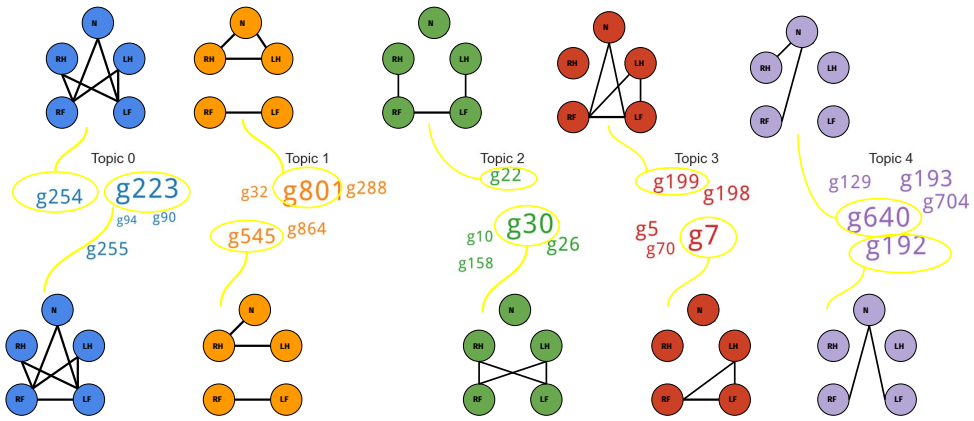


Figure 25: Visual representation of the five obtained topics described by their 5 most probable configurations: the top 2 are depicted as graphs whereas the last 3 are synthetized by their encoding. The size of a configuration encoding is proportional to its weight in topic-configurations probability distribution.

Table 8: Results of topic analysis. For each topic, we report statistics on: Intra-class assignment probability (mean, minimum, maximum, and concentrations), Symmetry (global, hands, and feet), and Density of the 5 most probable configurations.

	Intra-class probability								**Symmetry**			**Density**
	class N				**class Ab**							
	Mean	Min	Max	Conc	Mean	Min	Max	Conc	Global	Hands	Feet	
T0	0.59	0.34	0.9	0.15	0.67	0.6	0.91	0.15	0.94	0.95	0.78	0.62
T1	0.8	0.41	1.0	0.28	0.74	0.42	1.0	0.28	0.94	0.87	0.60	0.28
T2	0.73	0.46	1.0	0.17	0.74	0.36	1.0	0.13	0.90	0.80	0.75	0.34
T3	0.6	0.4	0.99	0.23	0.71	0.33	0.93	0.17	0.80	0.50	0.58	0.34
T4	0.7	0.44	0.98	0.17	0.82	0.47	1.0	0.26	0.92	0.20	0.68	0.24

4.3.4 *Discussion and conclusions*

The class-specific topic proportions analysis associates each infant to a predominant topic. The structural features that we computed (symmetry and density) attempt to reflect some qualitative aspects considered by experts physician during the motor evaluation [170]. Considering the results highlighted in Table 8, we can comment that:

1. the higher variability associated with infants with normal motion patterns is also a qualitative aspect that it is usually considered by expert physicians during their evaluations [170].

2. Infants with abnormal motion patterns are well represented by Topic 0 since the minimum probability is higher (0.60) with respect to the other cases. Topic 0 is also characterized by dense configurations and with a higher level of symmetry. Also in this case we have a correspondence between our results and the visual evaluation of expert physicians because abnormal movements are characterized by a higher symmetry.

3. Topic 4 is one of the two most frequent topic in infants with abnormal motion patterns and with the highest mean assignment probability. As for Topic 0, we can notice highly symmetric configurations. In this case the symmetry is quite entirely concentrated on feet connections.

For future works we plan to include more infants in the study and to refine the network representation detecting more landmark points on infants' bodies. By increasing configurations size, we expect to gain enough information to consolidate the analysis and investigate possible discriminative properties of the identified topics.

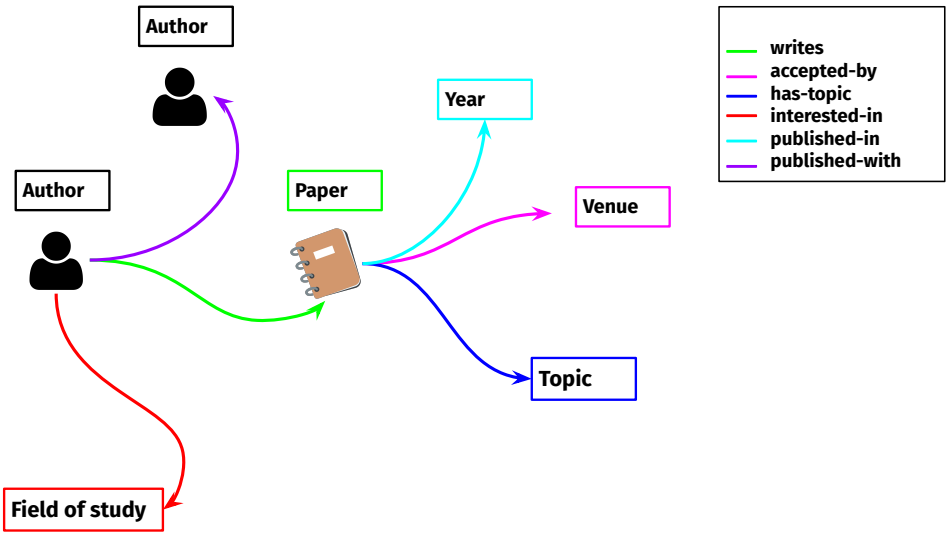


Figure 26: Academic Collaboration Knowledge Graph representation. Nodes represent authors, papers, fields of study, topics, venues and years of publication. Relationships among nodes identify different triplets as described in the top-right box of the figure.

4.4 Interactive recommendation in Academic Collaboration Networks.

Part of this section will appear in the following publication (to be submitted): *Garbarino, Davide, Giampaoli, Daniele, Cuneo, Marina, Paniati, Giorgia, Vian, Andrea, & Barla, Annalisa (ongoing). Interactive recommendation in Academic Collaboration Networks. In the Special Section on "Emerging Trends and Advances in Graph-based Methods and Applications", IEEE Transactions on Emerging Topics in Computing.*

It is known that collaboration between authors leads to a positive impact on research. This section aims to provide a preliminary analysis of the complex structure of a heterogeneous information network among researchers of the Machine Learning Genoa Center (MaLGa). We aim to examine the academic collaboration knowledge graph created starting from the data of the papers published by the MaLGa members during the period 1986–2022. We apply the main Network Analysis and Machine Learning techniques to describe the relational structure of the group of researchers and its evolution over time. As a concluding experiment, an attempt is made to classify papers based on the MaLGa research unit researchers belong to.

4.4.1 *Introduction*

Research centers are constantly monitored to evaluate scientific production, scientific collaboration between researchers, and the degree of openness compared to other national and international research centers [96]. Indeed, in recent years there has been an increase in scientific collaboration between researchers [126, 144], resulting in complex interactions involving actors working in the same disciplinary field, but also, and above all, actors from different fields of study. Collaboration could be analysed through co-authorship in scientific publications, as shown by several studies [91, 128, 220]. Nevertheless, recent advances in Heterogeneous Information Network (HIN) analysis [269] allow us to consider Academic Collaboration Networks as complex systems involving several types of entities (*e.g.* authors, papers, venues and fields of study) and several types of relationships among them (*e.g.* co-authorship, author-writes-paper, paper-accepted by-venue and so on and so forth).
Being able to represent a collaboration knowledge graph in such a way that semantics complexity is preserved may allow to perform tasks and consequently define usable systems able make the academic community even more connected. More specifically, heterogeneity in collaboration knowledge graphs may be leveraged to represent research groups (or more in general universities) by different perspectives (see Figure 26) and this would allow to recommend people, works or, more in general, research topics characterizing the group to external users (*e.g.* students looking for scholarships, or authors looking for collaboration on specific topics). To this end, Heterogeneous Graph Neural Networks (Section 3.3.4) [223, 255, 265] are a suitable tool that allow to represent nodes and edges of HINs by preserving their semantics while performing specified downstream tasks during optimization. In this framework, we would like to define a *social recommendation inductive learning problem*. Specifically, social recommendation aims to recommend relationships, in a network (either homogeneous or heterogeneous), to nodes which are not present in the system. This problem has been studied in depth for homogeneous networks [160, 165, 270], for which the goal translates into *link prediction* since there is only one type of nodes and edges, and for multiplex networks [245], in which nodes and relationships of different types appear at different layers.
Heterogeneous Graph Neural Networks have been widely exploited in social recommendation tasks [19, 250, 271] with some limitations. More specifically,

1. as pointed out in [131], academic collaboration networks are dynamic, meaning that they change over time. In particular, researchers-specific research interests evolve in time depending on many complex and usually latent factors. Thus, a social recommendation system must be able to incorporate this dynamic into researcher representations in order to make meaningful recommendations. However, present Heterogeneous GNN architectures hardly handle representations changing over time;

2. in academic collaboration networks, the nodes relating to textual features (*i.e.*, research interests and keywords extracted from the papers, the lat-

ter used as features of the papers nodes) play a fundamental role. The automatic extraction of keywords and the consequent representation is a complex task for various reasons. First of all, there exist NLP architectures (*e.g.* BERT [81]) that are able to extract keywords from different types of texts (for instance biomedical corpora [186]) but, to the best of our knowledge, there is no architecture trained on data science-related texts in the literature. Since data science is a highly multidisciplinary discipline, this problem is strongly exacerbated in our context. Furthermore, there is a semantic hierarchy underlying keywords. For example, supervised learning and machine learning are considered two distinct keywords belonging to the same context by current architectures, but the fact that machine learning is a much broader concept than supervised learning is not considered. Learning an ontology would then be an essential tool for social recommendation based on users specific research interests.

The work presented here is aimed to overcome the above mentioned issues related to social recommendation in academic collaboration networks by specializing the discussion of the topic to a case study, *i.e.* the collaboration network of the Machine Learning Genoa Center.
We are not able to offer a complete and concluded discussion of the work as it is in its early stages. However, given the relevance of the application with the arguments presented in this thesis, it seemed interesting to us to include a discussion of the work, albeit partial.

4.4.2 *the MaLGa center and dataset description*

The Machine Learning Genoa Center (MaLGa)[5] is a joint research center between computer science and mathematics. Its activities span a wide range of diverse but connected topics, including Computer Vision, Computational Harmonic Analysis, Data Science, Statistical Learning and Optimization.

MaLGa approach to research is highly interdisciplinary. Beyond pure research, knowledge transfer towards technological and industrial applications is a main objective of the center. Part of MaLGa mission are educational activities in machine learning and related topics, including scholarly-oriented teaching and professional training.
The MaLGa center is subdivided into four research units:

- **LCSL** unit: the Laboratory for Computational and Statistical Learning (LCSL) focuses on the development of efficient and reliable machine learning algorithms blending tools from statistics, optimization, and regularization theory and data science applications;
- **MLV** unit: the Machine Learning and Vision (MLV) unit investigates different nuances of visual perception in artificial intelligence systems,

5 https://malga.unige.it/

where computer vision and machine learning are combined to obtain robust data-driven methods addressing a variety of problems. In particular, MLV unit studies and develops methods for scene understanding, motion analysis, and action recognition, with applications to assisted living, human-machine interaction, and robotics;

- **CHarML** unit: the Computational Harmonic Analysis and Machine Learning (CHarML) unit focuses on connections among harmonic analysis, inverse problems, PDE and machine learning.
- **PiMLB** unit: the Physics informed Machine Learning for biological Behavior (PiMLB) unit blends physics, machine learning and biological behavior to ask how organisms acquire and process complex sensory information from their fluid environment to guide behavior.

The MaLGa center is currently composed by 59 researchers, comprising faculties, post-docs and Phd students. We collect data of the papers published by the MaLGa members during the period 1986–2022. Among a total number of 609 publications, we extract 311 papers equipped with keywords (obtained by venue keywords and the ones specified by authors) and labeled with a MaLGa unit class encoding (meaning a discrete label ranging from 1 to 4).
We build a heterogeneous MaLGa collaboration network as the one schematically described in Figure 26. In the MaLGa collaboration network nodes represent:

- *authors* (**A**), meaning MaLGa members and co-authors;
- *papers* (**P**) published by MaLGa members together with co-authors. For each paper we consider three types of node:
  - *topics* (**T**) identified by keywords collected from the venue the paper was published in and from the keywords attached by the authors;
  - the *year* (**Y**) of publication;
  - the *venue* (**V**) in which the paper was published.
- we link to each author several *field of study* (**F**) nodes, representing the collection of keywords from her/his works.

In the next section we present some analysis of the MaLGa collaboration graph to support the fact that the temporal information in textual data is a component that must be taken into account in order to model properly research interests.

4.4.3 *Analysis of MaLGa keywords*

In Figure 27 we schematically describe weighted keywords appearing in MaLGa published works at selected timepoints.
In 1986 the main interest concerns applications of data science to vision-related problems. In 1993, MaLGa members start to focus also on theoretical machine

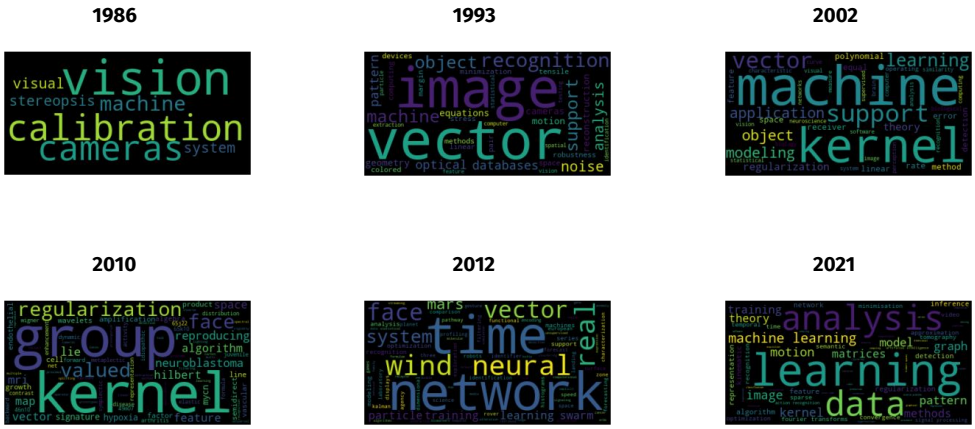


Figure 27: Wordcloud representation of research MaLGa interests at different timepoints. The size of each word appearing in each panels is proportional to its frequency in published works in that timepoint.

learning, classification problems and image processing. Starting from 2002, kernel and regularization methods become objects of interest in many publications. In more recent years, research interests begin to revolve around deep learning and graph data.
Such observations are further supported by the type of venue which accepted MaLGa works. Figure 28 describes the proportion of the total works presented at venues at different years. More recent years present an interdisciplinary scenario and this is mainly due to the fact that people having different scientific background recently entered the MaLGa center contributing with domain expertise and heterogeneous research interest.

4.4.4 *Papers classification as a proof of concept*

In order to test the representational power of Heterogeneous GNN architectures and the meaningfulness of relationships defined in the Knowledge Graph in Figure 26, we set up a nodes classification problem on paper nodes based on research unit labels, *i.e.* LCSL, CHarML, MLV and PiMLB (Section 4.4.2). The classification problem is extremely unbalanced. Indeed we have 80 papers in CHarML class, 162 papers in LCSL class, 61 papers in MLV group and 8 papers in PiMLB group. Such differences are due to the fact that data collection is still ongoing, conformations of groups change in time and PiMLB is a newly born research unit.

We tackle the classification problem by leveraging a 2-layer Relation-aware Heterogeneous Graph Neural Network [265] followed by a single-layer perceptron as the output layer. We split the 311 papers in the collected dataset into training, validation and test set each composed by 251, 28 nd 32 papers respectively. We instantiate MaLGa collaboration network nodes with a 256-

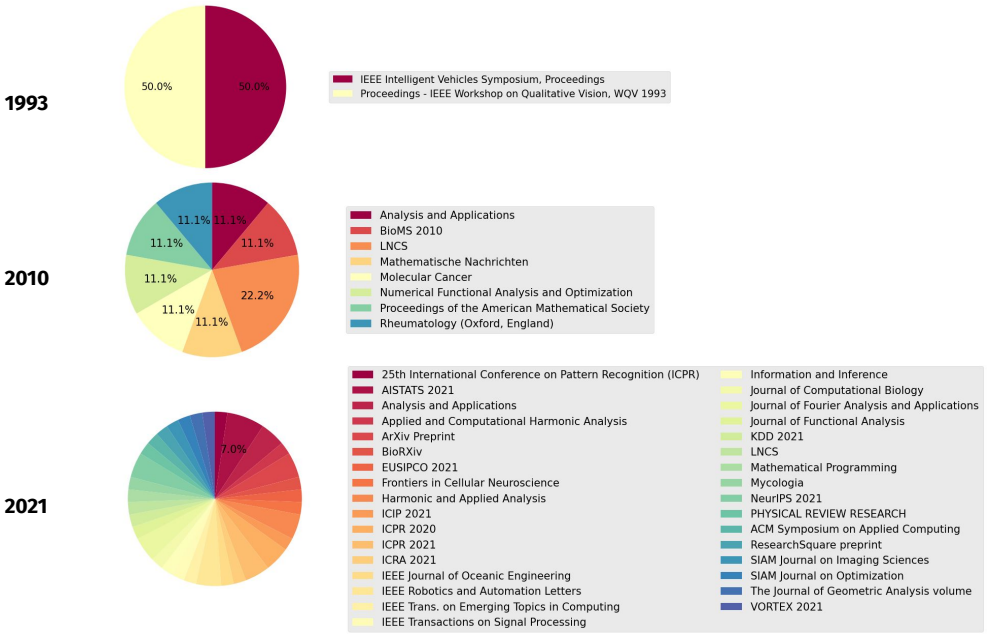


Figure 28: Pie charts representing venues that accepted MaLGa works at different timepoints. Last decades show a more heterogeneous and numerous set of venues due to the fact that several researchers joined the MaLGa center bringing in various expertise and interdisciplinary interests.

dimensional vector obtained by training different *metapath2vec* models based on specified metapaths:

- **APVPA**, which finds similar representations for authors publishing papers at the same venue;
- **AFA**, which relates authors who have similar research interests. A PCA visualization of such representation is presented in Figure 29.

EXPERIMENTAL SETUP The Relation-aware Heterogeneous Graph Neural Network model is optimized via the Adam [138] optimizer with the cosine annealing learning rate scheduler [157] and we use dropout [233] to prevent over-fitting. Since the proposed approach additionally captures the characteristics of relations, we set the hidden dimension of relation representation to 64. We set the number of heads in the multi-head attention mechanism to 8 and we train the model in a full-batch manner since it is a small-scale dataset. We implement the two-layer R-HGNN with PyTorch [197] and Deep Graph Library (DGL) [254].

RESULTS AND DISCUSSION We evaluate the multi-label classification problem through the accuracy metric (*i.e.* the number of correct predictions divided by the total number of examples) and the macro f1 score (*i.e.* the averaging of the precision and recall scores of individual classes). Macro-averaging the f1 score is a suitable approach in case of imbalanced classes because it weighs

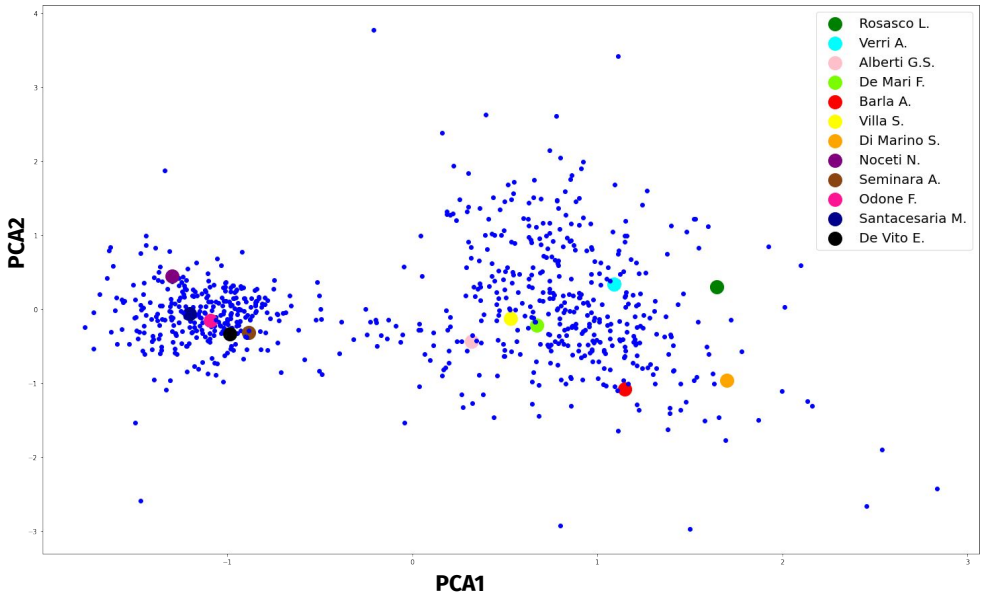


Figure 29: MaLGa researchers PCA visualization based on the metapath **AFA**. Bigger sized nodes correspond to MaLGa faculties, while other nodes are co-authors or other MaLGa members. Even if some ground truth similarities are actually recovered by the model, still the poor quality of textual information (*i.e.* the field of study node), does not allow to detect meaningful connections.

each of the classes equally and is not influenced by the number of examples of each class. Figure 30 shows the results obtained in the papers classification problem.

Although the performance of the classifier is good, the representation of papers does not seem to capture the commonality between the keywords characterizing them. This is quite evident from the Figure 31 in which papers appear to cluster more based on co-authorship semantics than similarity of their keywords.
As discussed in Section 4.4.3, for future works, a refined search and temporal analysis of keywords is required in order to really capture the complexity of the heterogeneous collaboration network.

	ACCURACY	MACRO f1
TRAINING	0.9960	0.9968
VALIDATION	1.0000	1.0000
TEST	0.9688	0.9701

Figure 30: Results of papers classification problem in terms of accuracy and macro f1 score. the results are encouraging in the sense that, although the data is poor, the representation obtained is able to discriminate the papers on the basis of macro research areas.

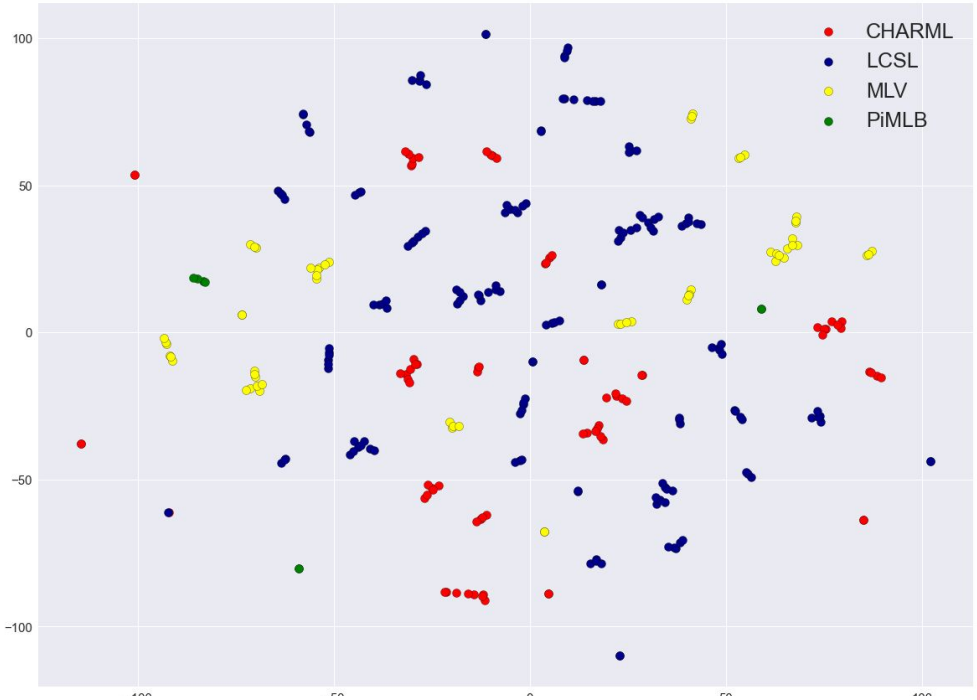


Figure 31: TSNE visualization of papers vector representation labeled by the true research unit they belong to as described in the legend.

PART II

Time-varying Gaussian Graphical Models

This second part of the thesis contains a description of the context and contributions about regularized network inference. Chapter 5 presents the steady-state regularised inference methods under the Gaussian distribution assumptions. Chapter 6 focuses on contributions on the concept of missing data in Gaussian Graphical Models, both at random or latent, and on the study of temporal Gaussian Graphical Models when data is non-stationary.

5

Regularized Markov Models

Markov Random Fields (MRFs) model conditional probability dependencies between variables. Consider the following example: given two genes A and B they are linked if, given the profiles of all other genes across all the subjects, the levels of genes A are still predictive for the gene B and vice versa. Therefore, a connection in a MRFs has a stronger meaning than correlation. MRFs are widely used in many applications as a mathematical abstraction of a system that allows to straightforwardly study its properties. Among the methods for the inference of MRFs from data we restrict our focus on those based on the assumption of sparsity, i.e., in high-dimensional contexts the connections that explain the state of the system are few with respect to the total number of possible edges between the variables. Guided by this prior, the inference of the graph depends on the probability distribution assumed on the data. In this thesis we will analyse one probability distribution: Gaussian, that allow to model continuous variables.

OUTLINE In this chapter we briefly introduce the concept of Markov Random Fields (Section **ref**). In Section 5.2 we explain how generalized linear models can be used to infer MRFs starting from those exponential family distributions that have linear sufficient statistics. In Section 5.3 we introduce the problem of penalised network inference. In Sections 5.4 we present Gaussian Graphical Models and in Section 5.5 their state-of-the-art extensions that consider time. We conclude in Section 5.5 with a brief summary of the chapter.

5.1 Markov Random Fields

MRFs are a set of models that belong to the wider set of *probabilistic graphical models*, which, beyond MRF, includes *Bayesian Networks*, *factor graphs* and *chain graphs* [69, 98, 147]. Probabilistic graphical models allows a graph to express the conditional dependence structure between random variables, *i.e.*, they define a joint probability distribution on a set of variables.

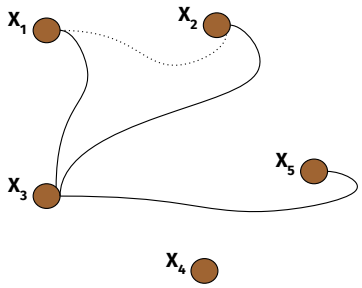


Figure 32: Example of a graphical model where the node X_2 is independent from the nodes X_1, X_5, and X_4 given the other nodes. The dashed edge represents the edge we condition away by considering the note X_3.

Consider a graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ where $\mathbf{V} = \{1, \ldots, D\}$ and $\mathbf{E} = \{(i, j) \mid i, j \in \mathbf{V}\} \subseteq \mathbf{V} \times \mathbf{V}$. The set $\mathbf{V}$ has a bijective correspondence to the set of variables $\{X_1, \ldots, X_D\}$ representing entities of the system in analysis. Then, a MRF is a probabilistic model that factorises according to a graph $\mathbf{G}$ in such a way that the conditional dependencies between the variables $X_1, \ldots, X_D$ can be directly read from the edges $\mathbf{E}$. Consider the graph in Figure 32. Here, the nodes X_2 and X_1 are independent given the node X_3, as no connections between them exists. If the node X_3 was not considered in the inference, we could not condition its presence away and thus, an edge between X_2 and X_1 would appear in the graph (dashed edge).

Definition 4 (Markov Random Field) *A Markov Random Field (MRF) is an undirected (possibly cyclic) graph over a set of random variables that satisfies the Markov property.*

The Markov properties are:

Definition 5 *Pairwise Markov property] Given two non-adjacent nodes u and v they are conditionally independent given all other variables: $u \perp\!\!\!\perp v | \boldsymbol{V} \setminus \{u, v\}$.*

Definition 6 (Local Markov property) *A variable u is conditionally independent of all other variables given its neighbours denoted by $\mathcal{N}(u)$: $u \perp\!\!\!\perp \boldsymbol{V} \setminus \mathcal{N}(u) | \mathcal{N}(u)$.*

Definition 7 (Global Markov property) *Any two sets of variables A and B are conditionally independent given a separating set S that contains all the paths connecting the nodes in A and B: $A \perp\!\!\!\perp B | S$.*

The properties grow in strength, however they are equivalent in the case of a positive probability [147].

5.1.1 *Gibbs Random Fields*

Definition 8 (Clique) *A clique C is a subset of vertices $\boldsymbol{V}$ of a graph $\boldsymbol{G}$ such that all the possible couples of nodes in the subset are adjacent,* i.e., *the corresponding sub-*

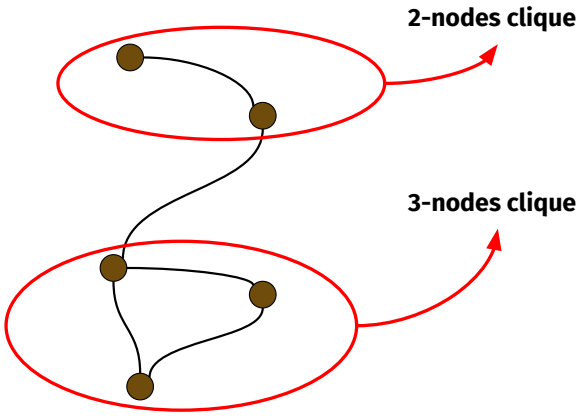


Figure 33: Example of 2-nodes and 3-nodes: small fully connected sub-graphs in an undirected graph of 5 nodes.

graph $\mathbf{G}_V$ is complete (see Figure 33 for a visual representation). A clique is maximal *if it is not properly contained within any other clique.*

The graph $\mathbf{G}$ completely defines a probability distribution $p_{\mathbf{G}}$ over the variables $X_1, \ldots, X_D$. In order to see this more clearly we recall the Hammersley-Clifford theorem [69], that states that when the probability $p_{\mathbf{G}}$ is strictly positive, a MRF is equal to a Gibbs random field. Therefore, it can be represented as a sum of functions on the graph's cliques. Given the graph $\mathbf{G}$ we can represent the related joint distribution $p_{\mathbf{G}}$ over the variables $X_1, \ldots, X_D$ as the product of compatibility functions that depend only on the subset of variables corresponding to its cliques. Let $\mathcal{C}$ be the set of cliques of the graph $\mathbf{G}$ and let $\{\phi_C(X_C), C \in \mathcal{C}\}$ be a set of clique-wise sufficient statistics that depends on the probability assumed on the data then, any distribution within the graphical model family represented by the graph $\mathbf{G}$, takes the form [69, 251]

$$p_{\mathbf{G}}(X_1, \ldots, X_D) \propto \exp\left(\sum_{C \in \mathcal{C}} \theta_C \phi_C(X_C)\right) \tag{39}$$

Then, given the set $\mathcal{C}$ of cliques of the graph, an MRF is a collection of distributions that factorises as

$$p_{\mathbf{G}}(X_1, \ldots, X_D) = \frac{1}{A} \prod_{C \in \mathcal{C}} \exp\left(\theta_C \phi_C(X_C)\right) \tag{40}$$

where A is a log-normalisation constant chosen to ensure that the distribution sums up to 1 [251]. Note that the use of all cliques may be a redundant definition but allows for easier computation while not yielding to loss of generality [251]. Often, it may be useful to use the direct factorisation of the joint probability, defined as

$$p_{\mathbf{G}}(X_1, \ldots, X_D) = \prod_{v \in \mathbf{V}} p(X_v | X_{\mathcal{N}(v)}) \tag{41}$$

where $\mathcal{N}(v)$ is the set of variables in the neighbourhood of the variable v.

5.2 Markov Random Fields and the Exponential Families

Exponential families can naturally be interpreted as probabilistic graphical models and more specifically Markov Random Fields. This is due to the fact that exponential families are represented as the summation of weighted functions similarly to the form in Equation 39.

5.2.1 *Exponential Families*

Exponential families are a very flexible family of distributions that includes many of the most known distribution such as Bernoulli, Gaussian, Multinomial, Dirichlet, Gamma, Poisson and Beta.

Definition 9 (Exponential family) *Given a sample space $\mathcal{X}^D$ on which it is defined a measure ν and random vector $(X_1, \ldots, X_D) \in \mathcal{X}^D$ we define a collection of functions $\phi = (\phi_\alpha : \mathcal{X}^D \longrightarrow \mathbb{R})_{\alpha \in \mathcal{I}}$ called sufficient statistics to which we associate their exponential parameters $\theta = (\theta_\alpha)_{\alpha \in \mathcal{I}}$. Then, the associated exponential family is defined as the following parameterised collection of density functions*

$$p_\theta = (X_1, \ldots, X_D | \theta) = \exp\Big(\theta, \phi(X)\rangle + h(X) - A(\theta)\Big) \tag{42}$$

where $h(X)$ is a function of only the samples and $A(\theta)$ is the log-normalisation function that ensures the probability to be properly normalised and it is defined a

$$A(\theta) = \log \int_{\mathcal{X}^D} \exp\Big(\langle \theta, \phi(X)\rangle\Big) \nu(dX). \tag{43}$$

By fixing the sufficient statistics ϕ we identify a particular type of exponential family (*e.g.*, Poisson or Bernoulli). By also fixing the exponential parameters θ we define a specific member of such family (*i.e.*, a specific probability distribution).

5.2.2 *Exponential-family based Graphical Models*

It is possible to reason in terms of MRFs for any exponential family distribution [251, 260, 261]. This representation is particularly suited when the sufficient statistics are linear in the variables as it allows to obtain a straightforward inference algorithm. Suppose we are given a univariate exponential family distribution

$$p(X) = \exp\Big(\langle \theta, \phi(X)\rangle + h(X) - D(\theta)\Big) \tag{44}$$

with log normalisation function $D(\theta)$.
Consider a D-dimensional random vector $X = (X_1, \ldots, X_D)$ and an undirected

graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ over D variables. Suppose now that the distribution on the variable X_v given the rest of the nodes X_{-v} has the form in Equation 40 with sufficient statistics $\{\phi(X_s)\}_{s \in \mathcal{N}(v)}$. Then such distribution is a linear combination of k-th order products of univariate functions

$$\begin{aligned} p(X_1, \dots, X_D|\theta) = \exp\Bigg(& \sum_s \theta_s \phi(X_v) + \sum v \in \mathbf{V} \sum_{s \in \mathcal{N}(v)} \theta_{vs} \phi(X_s) \\ & + \sum_{v \in \mathbf{V}} \sum_{s_2, s_3 \in \mathcal{N}(v)} \theta_{v s_2 s_3} \phi(X_2) \phi(X_3) \\ & + \sum_{v \in \mathbf{V}} \sum_{s_2, \dots, s_k \in \mathcal{N}(v)} \theta_{v s_2 \dots s_k} \prod_{j=2}^{k} \phi(X_{s_j}) + h(X_v) - A(\theta) \Bigg). \end{aligned} \tag{45}$$

Then, under the assumptions:

1. the joint distribution factorise according to a graph $\mathbf{G}$ which has clique factors of size at most k;
2. the node-conditional distribution follows an exponential family;

the conditional and joint distributions are given by Equation 44 and 45 respectively [34, 69, 251, 260, 261].
This strictly connects exponential family with MRFs as the exponential parameters are nothing else but the connections of cliques of different size in the graph.

When the joint distribution has factors of size at most two ($k = 2$) and the. sufficient statistics are linear functions $\phi(X_v) = X_v$ than the conditional distribution is a generalized linear model [185] with conditional distribution of the form

$$p(X_v|X_{-v}) = \exp\left(\theta_v X_v + \sum_{s \in \mathcal{N}(v)} \theta_{vs} X_v X_s + h(X_v) - \bar{D}(X_{-v}, \theta) \right) \tag{46}$$

and joint distribution

$$p(X_1, \dots, X_D|\theta) = \exp\left(\sum_v \theta_v X_v + \sum_{(v,s) \in \mathbf{E}} \theta_{vs} X_v X_s + \sum_{X_v} h(X_v) - A(\theta) \right). \tag{47}$$

5.3 Network Inference

Network inference or graphical model selection aims at selecting the most probable graph from observations of the variables. It arises in lots of applications when the underlying graph structure of variables is not known [24]. Suppose to have D random variables denoted $X_1, \dots, X_D$ of which we can observe N samples $X \in \mathcal{X}^{N \times D}$ where $X[i,:] = (X[i1], \dots, X[iD])$ for $i = 1, \dots, N$. We aim at inferring the set $\mathbf{E}$ of edges of the graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, that better fits the data.

Such goal can be reached with inference methods based on Maximum Likelihood Estimation (MLE) . The concept of MLE relies on the maximization of a likelihood function of the model. Indeed, such value tells us how much observations are likely given a model defined by a set of parameters. In our case the model corresponds to the graph $\mathbf{G}$, whose parameters θ we represent by its adjacency matrix K.

Definition 10 (Likelihood) *Given data X and the parameters K the likelihood $L(X|K)$ of the graph $\mathbf{G}$ is any function of K proportional to the density function $p(X|K)$.*

Note that the likelihood is a function of the parameter K for fixed X whereas the probability is a function of X for fixed K. It is common, for optimisation problems, to use the log-likelihood $\ell(X|K)$ (the natural logarithm of the likelihood function $L(X|K)$) as it allows to remove products and exponentials within the function to optimise, plus avoiding numerical issues when $D \gg N$.

Suppose now that, for fixed data X, we have two possible values for the parameters K, K' and K'', and that $L(X|K') \geq L(X|K'')$. This means that K' is at least likely as K'' and, therefore, it is the one the better supports the data. This naturally leads to the concept of Maximum Likelihood:

Definition 11 (Maximum Likelihood) *A maximum likelihood estimate of K is a value, K^*, that maximizes the likelihood $L(X|K)$ — or the log-likelihood $\ell(X|K)$.*

5.3.1 ℓ_1 *penalisation*

The inference of graphical models through MLE still remains a difficult problem given the dimension of the possible search space. If we consider D variables, it is combinatorial in the number of possible edges, $2^{D\frac{D-1}{2}}$. We can reduce the search space by assuming *sparsity* of the solution. Such assumption, by constraining the problem, eases the identification of the graph, improves interpretability of the results and reduces the noise. It is fundamental especially when the number of variables is higher than the number of available samples (the so-called $D \gg N$ problem).
Formally, this translates into the addition to a MLE problem of a sparsityenforcing penalty called ℓ_1-norm. Such norm is a convex non-smooth function that is often used as a relaxation of the non-convex ℓ_0-norm that enforces the number of edges to be small. Given the adjacency matrix K of the graph G, the ℓ_1-norm is defined as

$$||K||_{1,od} = \sum_{ij, i \neq j} |K_{ij}| \tag{48}$$

Such norm penalises the weight of edges between the variables shrinking their value and forcing those edges that have values in an interval $[\alpha, \alpha]$ to be zero, thus selecting only a subset of possible connections. Here, α measures the strength of the penalty on the problem, the higher the α the higher the number of zero edges. Such penalisation approach has been widely used in literature,

and the very first model exploiting such idea was proposed and developed by [171] for a neighbourhood estimation of Gaussian Graphical Models.

Definition 12 (Neighborhood Estimation) *Penalised neighbourhood estimation infers the conditional independence separately for each node in the graph solving a lasso-like problem where the considered variable is the dependent variable and the others are considered as independent covariates.*

Consistency proofs of such approach are provided [171], and they can be extended for logistic regression [252]. In particular, in [252] they provide sufficient conditions on the number of samples, dimensions and neighbourhood size to estimate the neighbourhood of each node simultaneously. Neighbourhood estimation, nevertheless, has been shown to be an approximation of the exact problem [100, 267] as it does not yield to the MLE when there is no equality between the (possibly perturbed) empirical covariance matrix and the estimated one. In [100] the authors bridge the conceptual gap between this and the exact problem proposing the graphical lasso method, based on the work of [20]. Later, the concept of an ℓ_1 penalised MLE was proposed also for non Gaussian distributions [20]. Again inference is performed via neighbourhood selection as the computation of the joint likelihood is infeasible [39, 171, 211, 212, 260, 267].

5.4 Gaussian Graphical Models

Gaussian Graphical Models (GGMs) are widely used, for example in psychology [73, 86], biology [136, 235] and neurology [230] and they are particularly suited for the modelling of continuous variables. This means that the sample space is defined as $\mathcal{X} = \mathbb{R}$. GGMs are probabilistic graphical models which variables are jointly distributed according to a multivariate Gaussian distribution $\mathcal{N}(\mu, \Sigma)$ where $\mu \in \mathbb{R}^D$ is the mean vector, and $\Sigma \in \mathbb{S}^D_{++}$ is the $D \times D$ covariance matrix. For simplicity, unless otherwise specified, throughout this thesis the normal distributions are assumed to be centred in 0, *i.e.*, $_i = 0$ $\forall i = 1, \dots, D$. This is assumed without loss of generality as we let the variables to be completely explained by the covariance matrix Σ [65].
If Σ is a well-defined covariance matrix, (*i.e.*, is positive definite), then the conditional independence between variables in the multivariate normal distribution is associated to zero entries in its inverse [77].

Proposition 1 *Let $X \sim \mathcal{N}(0, \Sigma)$ be a random vector drawn from a multivariate normal distribution, where $K = \Sigma^{-1}$ is the precision matrix of the distribution. Let Γ be the set of entries in Σ. Then, for each $v, s \in \Gamma$ with $v \neq s$,*

$$X_v \perp\!\!\!\perp X_s | \Gamma \setminus \{v, s\} \implies K[vs] = 0. \tag{49}$$

This result follows from standard linear algebra, details and proof of the proposition can be found in [147]. Therefore, the precision matrix is associated to the graph $\mathbf{G}$, where an edge exists if and only if the two variables have a

value different than zero in the corresponding entry of the precision matrix K. For this reason, the precision matrix can be considered as the weighted adjacency matrix of $\mathbf{G}$, encoding the conditional dependences between variables. The Gaussian distribution is the only exponential family distribution for which it is feasible to compute the normalisation constant. Indeed, such value is defined as the integral over all the possible values of the random variables, and, only with the Gaussian distribution, we can compute a finite form [251]. Therefore, this allows to reason in terms of joint distribution which leads to more consistent results [100]. Let $X = (X_1, \dots, X_D) \sim \mathcal{N}(0, \Sigma)$ indicate a random vector, and X the dataset containing N realisations of the D variables in such a way that $X \in \mathbb{R}^{N \times D}$, then the density function is defined as

$$p(X|\Sigma) = \frac{1}{2\pi^{\frac{D}{2}} \det(\Sigma)^{\frac{1}{2}}} \exp\left(-\frac{1}{2} X\Sigma^{-1}X^T \right). \tag{50}$$

Given the set of N iid samples in D dimensions in matrix X, based on Equation 50, the Gaussian log-likelihood is defined as

$$\ell(X|\Sigma) = \log \prod_{i=1}^{N} p(X_i|\Sigma) = -\frac{N}{2} \log\det(\Sigma) - \frac{N}{2} \mathrm{tr}(\frac{1}{N} X^T X\Sigma^{-1}). \tag{51}$$

This likelihood is expressed in function of Σ but, in literature, it has been shown that estimating the precision matrix K leads to better results [20, 39, 100, 147, 171, 212, 252, 267]. We re-write the likelihood in terms of the precision matrix K as

$$\ell_{GGM}(X|K) \propto N \log\det(K) - \mathrm{tr}(\frac{1}{N} X^T XK) + c, \tag{52}$$

where $\log\det$ denotes the logarithm of the determinant of the matrix K, tr is the trace function defined as $\mathrm{tr}(\cdot) = \sum_i (\cdot)[ii]$, *i.e.*, the sum of the diagonal elements of the matrix and c is a constant term.

5.4.1 *Lasso penalisation*

Given the sparsity assumption we want to force some entries of the precision matrix K to be zero, as introduced in Section 5.3.1 [171]. A model for the inference of K including the sparse prior is the graphical lasso (GL) what writes out as [100, 122]:

$$\text{minimize}_K - \ell_{GGM}(K|X) + \alpha ||K||_{od,1}, \tag{53}$$

where $||\cdot||_{od,1}$ is the off-diagonal ℓ_1 norm. Equation 53 has a lasso-like form [240]
. For this reason, the problem can be solved by coordinate descent, using a modified lasso regression on each variable in turn, thus leading to a simple, efficient and fast procedure.

5.5 Temporal Extensions

Problems in Equation 53 aims at recovering the structure of the system at fixed time (static network inference). However, complex systems may have temporal dynamics that regulate their overall functioning [7]. Hence, the modelling of such complex systems requires a dynamical network inference, where the states of the network are intended as co-dependent. Indeed, the analysis of a set of variables which describe the system at a particular time point could not provide enough information on the more global and general behaviour of the system. As an example, one may consider the analysis of genes observations under the presence of a particular phenotype. Static network inference would answer to the question regarding a particular status of the cell. The answer to the same question asked later in time could lead to a different answer. The idea of time-varying network inference is to continue the inference process in time. It could be seen as a generalisation of a static inference process that infers separately networks at different point in time. The addition is that time-varying network inference exploits the temporal component during the optimisation. This can improve performances as, in static network inference, there is no theoretical guarantee that the network at step t would be similar to the network at step $t+1$, while one may intuitively expect so. Dynamic network inference instead will embed prior knowledge on the evolution of the network which could help in presence of noise in particular time points of the network. Indeed, changes in the network at a particular time point may be due to external perturbation, noise or a particular developing state of the system. The dynamism can be modelled in different ways:

1. by assuming a specific temporal dynamic modelled by differential equations [2];
2. by assuming stochasticity on the edge of the networks [102, 200];
3. by assuming a temporal consistency modelled through a similarity function between contiguous time points [37, 112].

The first and second options are suitable for many applications but they, in turn, require a wide knowledge on the applicative domain. Temporal consistency, instead, allows us to be broad on the possible applications. In this thesis, as we do not have a specific domain in mind, we focus and exploit the concept of temporal consistency.

5.5.1 *Temporal Consistency*

In order to exploit temporal consistency we need to assume that the network models a non-stationary distribution that may change at each time point. This implies that to different time points correspond different states of the system that cannot be expressed by a unique model. We assume a consistency (or similarity) between consecutive states of the network, as, for sufficiently close

time points, a system would show negligible differences [112]. The same consistency principle can be exploit in other contexts as multi-class [74, 109] or multi-layer[61] network inference.

Definition 13 (Consistency) *Two inferred networks are said to be exactly consistent if the distance between the related network structures, in terms of some norm, is zero.*

The more the distance grows the less consistent the networks are. The inference of a dynamical network that assumes temporal consistency of consecutive time points can be performed through a regularised approach that extends the stationary model with the imposition of a penalty [103]. The main example of this type of dynamical inference is the context of GGMs and it is the *time-varying graphical lasso* (TVGL) [112] where the inference of a network at a single time point t is guided by the states at adjacent time points. Throughout the thesis this model will be also referred to as TGL. When we mention it as TVGL we intend the original version proposed in [112] while when we call it TGL we intend our re-implementation of such model. We will try to explicitly mention this when there is an ambiguity.
Consider now a system formed by D entities measured over T time points. For each time point t we have N_t samples randomly drawn as

$$X = (X_1, \dots, X_T) \sim (\mathcal{N}(0, \Sigma_1), \dots, (N)0, \Sigma_T))$$

where $X_t \in \mathbb{R}^{N_t \times D}$ for $t = 1, \dots, T$.
The goal, since we are in the Gaussian case, is to infer the precision matrices $K = (K_1, \dots, K_T) \in \mathbb{R}^{(D \times D) \times T}$ that encode the conditional dependencies at each time point [112]. The TVGL problem is defined as follows

$$\text{minimize}_{K \in \mathbb{S}^D_{++}} \sum_{t=1}^{T} -N_t \ell_{GGM}(X_t|K_t) + \alpha ||K_t||_{od,1} + \beta \sum_{t=1}^{T-1} \Psi(K_{t+1} - K_t), \quad (54)$$

where Ψ is a function that encodes prior information on the temporal behaviour of the network. The related parameter β imposes a certain strength on the consistency of such behaviour in time.

PENALTY FUNCTIONS [112] proposed different consistency functions that guarantee a fast optimisation of the related problem. In particular, we can choose among:

- $\Psi(\cdot) = \ell_1(\cdot) = \sum_{i,j} |\cdot|$, which is the lasso penalty that encourages few edges to change between subsequent time points while the rest of the structure remains the same [74];
- $\Psi(\cdot) = \ell_{12}(\cdot) = \sum_{j} ||\cdot_j||^2$, which is the group lasso penalty that encourages the graph to restructure at some time points and to stay stable in others [103, 111];
- $\Psi(\cdot) = \ell_2^2(\cdot) = \sum_{i,j} (\cdot_{i,j})^2$, which is the Laplacian penalty which encourages smooth transitions over time, for slow changes of the global structure [257];

- $\Psi(\cdot) = \ell_\infty(\cdot) = \sum_j (\max_i |\cdot_{i,j}|)$, which is the max norm penalty which encourages a block of nodes to change their structure with no additional penalty with respect to the change of a single edge among such nodes;
- $\Psi(\cdot) = \min_{V:A=V+V^T} \sum_j ||V_j||_p$, which is the row-column overlap penalty that encourages a major change of the network at a specific time, while the rest of the system is enforced to remain constant. Choosing $p = 2$ causes the penalty to be node-based, *i.e.*, the penalty allows for a perturbation of a restricted number of nodes [178]

5.6 Conclusion

In this chapter we provided background on graphical models inference methods based on the Gaussian probability distribution assumption. In particular, we showed how it can be possible to exploit a penalised Maximum Likelihood Estimation strategy to infer a graphical model that assumes a distribution belonging to the Exponential Family class. We described in details the inference for Gaussian Graphical Models and we discussed the state-of-the-art temporal extensions to these models focusing on the one that extends GGMs with temporal consistency.

6

Contributions

6.1 Missing Data in Gaussian Graphical Models

Part of this section is present in the following publication: *Tozzo, Veronica, Garbarino, Davide & Barla, Annalisa. Missing Values in Multiple Joint Inference of Gaussian Graphical Models. In Proceedings of the 10th International Conference on Probabilistic Graphical Models, PMLR 138:497-508, 2020.*

Real-world phenomena are often not fully measured or completely observable, raising the so-called *missing data* problem.

As a consequence, the need of developing ad-hoc techniques that cope with such issue arises in many inference contexts.

We focus on the inference of Gaussian Graphical Models (GGMs) from multiple input datasets having complex relationships (*e.g.* multi-class or temporal). We propose a method that generalises state-of-the-art approaches to the inference of both multi-class and temporal GGMs while naturally dealing with two types of missing data: partial and latent. Synthetic experiments show that our performance is better than state-of-the-art. In particular, we compared results with single network inference methods that suitably deal with missing data, and multiple joint network inference methods coupled with standard pre-processing techniques (*e.g.* imputing). When dealing with fully observed datasets our method analytically reduces to state-of-the-art approaches providing a good alternative as our implementation reaches convergence in shorter or comparable time. Finally, we show that properly addressing the missing data problem in a multi-class real-world example, allows us to discover interesting varying patterns.

6.1.1 *Introduction*

Consider the two following scenarios: (i) during a medical trial we submit a survey to patients to assess their status, for privacy concerns they refuse to answer to some questions; (ii) during a weather monitoring experiment, we

consistently do not measure the humidity in the air. These two types of situations lead to two different concepts: in the first case, questionnaires present missing values randomly positioned depending on the concerns of the patient, we call the final data *partial*. Differently, in the second scenario, we do not have any observation of humidity, thus we say that the related data are *latent* [154]. We will generically refer to these types of data as *missing data*, to indicate one of the two or a combination of both.

Missing data require careful consideration in all machine learning and statistical settings. We restrict to the problem of inferring multiple joint Gaussian Graphical Models (GGMs) when the input data may contain missing values. A multiple joint GGM is represented by a set of undirected graph $\mathbf{G} = (\mathbf{G}_k)_{k=1}^K = (\mathbf{V}, \mathbf{E}_k)_{k=1}^K$, where $\mathbf{V} = \{X_1, \dots, X_D\}$ is a finite set of nodes that represent random variables, and each $\mathbf{E}_k \subseteq \mathbf{V} \times \mathbf{V}$ is a set of edges, where $\mathbf{E}_k$ is not necessarily equal to $\mathbf{E}_j$ for $k \neq j$. Each $k = 1, \dots, K$ defines a specific connectivity pattern and its meaning depends on the context. Either it could be associated with K different classes when dealing with multi-class network inference [74, 109] , or it could index a sorted sequence of discrete time-points when facing a problem of temporal network inference [112, 180, 242]. Each $\mathbf{E}_k$ univocally determines a multivariate normal distribution $\mathcal{N}(\mu_k, \Sigma_k)$. Indeed, the *precision matrix*, $\Theta_k = \Sigma_k^{-1}$ encodes the conditional independence between pairs of variables, *i.e.*, the structure of the graph $\mathbf{G}_k$, since $\Theta_k(i, j)$ is 0 if and only if $(x_i, x_j) \notin \mathbf{E}_k$ [147] Therefore, one can interpret the precision matrix as the weighted adjacency matrix of $\mathbf{G}_k$.

Typically the graph structure $\mathbf{G}_k$ is not known as we can only observe the behaviour of the single variables in the system. We indicate the n_k observations of the D variables as $X_k = (x_{i1}, \dots, x_{iD})_{i=1}^{n_k} \in \mathbb{R}^{n_k \times D}$ which is a dataset that may contain missing values. Joint multiple network inference aims at learning a series of precision matrices $\mathbf{\Theta} = (\Theta_k)_{1 \leq k \leq K}$ that are both sparse and consistent with each other from these observations. The sparsity assumption is due to the combinatorial nature of the problem that requires to be constraint to have identifiability guarantees [100]. The consistency assumption fulfills the need of a similar structure among classes/time points that belong to the same underlying system [74, 97, 109, 112, 180, 242].

In literature, the problem of inferring graphs from observations that contain missing data has been tackled in the single graph inference case. A non-convex method based on Expectation Maximization (EM)[78] has been used to cope with the problem of partial data [154, 234] and latent data [266]. Differently, in [54, 55, 65] they considered a convex sparse plus low-rank decomposition method to marginalise out the effect of latent data. The inference of multiple joint GGMs in presence of latent data was tackled in [56, 97, 242] using the sparse plus low-rank approach only in the case of temporal networks. To the best of our knowledge, a unified way of dealing with all types of missing data and types of joint inference does not exist.

Here, we provide a method that is able to deal with both partial and latent data in the case of multiple GGMs. The novelty of the approach consists in providing a unique way of handling different types of multiple joint inference

methods as well as two types of missing data. Moreover, in the case of latent data, the method can provide more insights on the actual latent connections compared to previously published work. The method leverages on the EM algorithm to deal with missing data [234, 266] and kernels to deal with different types of joint inference problems. Indeed, in [241] they proposed a kernel method that, by suitable choices of the kernel, allows to consider simultaneously different types of complex temporal relationships and multi-class data. Synthetic experiments show that our method provides good results in terms of estimate of the true underlying network as well as good estimate of the true values of the edges. We thoroughly evaluated the proposed method under several levels of complexity both in the multi-class and the temporal case. We also show that our method is able to retrieve possibly better results than the state-of-the-art in case of complete data. We also provide a real-world case study in which we compare the performances of the newly proposed method in a multi-class setting. We show that, differently from the state-of-the-art method coupled with imputing techniques we are indeed able to capture variability patterns within the classes that are consistent with the data.

6.1.2 *Missing data*

Missing data are values of a dataset that are un-observed and may possibly be meaningful for a specific analysis task. We consider two possible patterns of data missing at random, *i.e.* the missing values does not depend on the value itself. Each of these two types of data introduces different problems during the inference of networks [154]. Given k data matrices $X_k \in \mathbb{R}^{n_k \times D}$ which samples are assumed to be drawn from a multivariate Gaussian distribution, we formally define the two types of missing data as follows.

Partial data can be described as the absence of measurements randomly positioned in each observation. More formally, partial data consist in a set of matrices X_k where for each $k = 1, \dots, K$ and for each $i = 1, \dots, n_k$ there are some unobserved values indexed by the set $M_i^k \subseteq \{1, \dots, D\}$. These "holes" in the matrix make impossible direct computation and thus require pre-processing or ad-hoc inference mechanisms. The pre-processing approaches are typically: *complete cases* and *imputing*. In the complete cases we discard the samples that do not have complete measurements on the variables, while *imputing* filles the holes with a suitable value, *e.g.* the empirical mean. The issues deriving from complete cases and imputing pre-processing are the following: the former reduces the sample size drastically which may impede the correct inference of the underlying graph especially when $n \ll D$; the latter introduces substantial bias in the estimated solution even if it induces to believe that we can reason in terms of complete data [154, 164]. Hence, pre-processing techniques distort the empirical distribution of the variables which consequently leads to biased estimates of the underlying graph.

Latent data describe the consistent absence of some variable measurements across all samples. More formally, we observe D variables but there are l more, indexed by the set $M_i^k = \{D+1, \dots, D+l\}$ that are always unobserved across

all data matrices $k = 1, \dots, K$ and samples $i = 1, \dots, n_k$. Therefore, on these variables, we do not have any information, not their number nor the relationship they have with the observed variables. Their presence in the system though, if not taken into account, leads to spurious edges, *i.e.*, links that would be conditioned away if the latent variables could have been observed [54].

6.1.3 *Joint multiple network inference*

We consider a multiple joint Gaussian Graphical Model defined as a set of multivariate normal distributions $\mathcal{N}_{\mathbf{G}}$ that factorises according to a set of graphs $\mathbf{G} = (\mathbf{V}, \mathbf{E}_k)_{k=1}^K$. The inference of such graphs corresponds to the learning of the precision matrices $\mathbf{\Theta} = (\Theta_1, \dots, \Theta_k)$ that completely define the underlying distributions assuming that the means μ_k is zero for every k. The index k can have different meanings depending on the context. We deal with the inference from: **multi-class data**, or *Joint Graphical Lasso* (JGL) [74, 109] where k indexes different classes of observations; **temporal data**, or *Time-varying Graphical Lasso* (TGL) [112, 242]. where k corresponds to discrete time points obtained by dividing a time-series of length T in K chunks of equal size. In each chunk the samples are assumed to be *i.i.d*. The concept of joint inference lies in the fact that we guide the inference method with the prior that there is structural consistency among the k precision matrices $\mathbf{\Theta} = (\Theta_1, \dots, \Theta_k)$. In order to deal with both JGL and TGL with a unique inference method we recur to the methodology proposed in [241] where the authors exploit a kernel formulation to model the dependencies allowing to consider in a single way different joint inference problems. A kernel $\kappa \in \mathcal{S}_+^T$ is a positive semi-definite matrix that encodes, at each entry $\kappa(k, k')$, the strength of how much two different networks, Θ_k and $\Theta_{k'}$, should be similar in such a way that, samples belonging to different (but related) input matrices, can drive the inference toward a more reliable estimation of the structure especially when the number of samples is low.

Consider two networks indexed by k and k', the kernel models similarities dependency strength in such a way that a strength equal to zero ($\kappa[k, k'] = 0$) implies that graphs $\mathbf{G}_k$ and $\mathbf{G}_{k'}$ are independent from each other and, therefore, no consistency is forced on them during the inference. In particular we will use for

- **multi-class data**, a kernel which enforces the same similarity on all classes

$$\kappa(k, k') = \begin{cases} 1, & \text{if } k = k' \\ \beta, & \text{if } k \neq k' \end{cases} \tag{55}$$

- **temporal data**, a kernel which enforces similarity only on consecutive time points

$$\kappa(t_i, t_j) = \begin{cases} 1, & \text{if } t_i = t_j \\ \beta, & \text{if } t_i = t_{j+1} \text{ or } t_i = t_{j-1} \\ 0, & \text{otherwise} \end{cases} \tag{56}$$

Here, $\beta > 0$ is a constant value that measures the strength of how similar the graphs k and k' are.

In order to impose structure similarity that derives from joint inference, we adopt a penalised Maximum Likelihood Estimation (MLE) method where the penalisation is given by [241]

$$P_{\Psi,\kappa}(\boldsymbol{\Theta}) = \sum_{\substack{s,k=1 \\ s>k}}^{K} \kappa_{sk}\Psi(\Theta_s - \Theta_k) = \sum_{k=1}^{K-1}\sum_{k'=1}^{K-k} \kappa_{kk'}\Psi(\Theta_{k+k'} - \Theta_{k'}). \tag{57}$$

Such penalty depends on the kernel κ that encodes the type of multiple joint network inference and the consistency function Ψ that defines the type of similarity to enforce on graphs that are dependent to each other according to the specified kernel.

In [112] the authors proposed different consistency functions. We make explicit use of the following two, but all the ones proposed in the original paper can be easily substituted. In particular, we will consider $\Psi(\cdot) = \ell_1(\cdot) = \sum_{ij} |\cdot|$, which is the lasso penalty that encourages few edges to change between subsequent time points while the rest of the structure remains the same [74] and $\Psi(\cdot) = \ell_2^2(\cdot) = \sum_{ij} \cdot_j^2$, which is the Laplacian penalty that causes smooth transitions [103, 111].

Note that substituting the explicit formulation of the kernels in Equations (55) and (56) in the penalty in Equation (57), leads to the original inference problems JGL [74] and TGL [112], respectively.

6.1.4 *Joint network inference with missing data*

For each k, consider a set of n_k observations $X_k \in \mathbb{R}^{n_k \times D}$ where each sample is a D-dimensional vector drawn from a multivariate Gaussian distribution $\mathcal{N}(\mu_k, \Theta_k)$ that may be not complete, *i.e.*, some values may be missing. We want to define a general network inference method that, from such observations, learns a set of precision matrices $\boldsymbol{\Theta} = (\Theta_1, \dots, \Theta_K) \in \mathbb{R}^{(D\times D)\times K}$ and means $\boldsymbol{\mu} = (\mu_1, \dots, \mu_K) \in \mathbb{R}^{D\times K}$. Differently from literature, we need to estimate the means instead of assuming data to be centered in zero. Indeed, in presence of partial data we cannot compute the empirical mean and re-centre the data without introducing bias as we would distort the empirical distribution in the same way as imputation does [154].

The modelling and the inference of GGMs from these type of data is aid by the factorisation properties that hold for the multivariate normal distribution. Consider, for each k and $i = 1, \dots, n_k$ the sets O_i^k and M_i^k of indices of observed and missing variables, respectively. Such sets allows us to divide the sample i as $X_k[i,:] = \left(X_k[i, O_i^k], X_k[i, M_i^k]\right)$. Accordingly, for each sample i it is possible to define block precision matrices that group the set of observed and missing variables together. This grouping procedure allows us to obtain a conditional distribution that is still a multivariate normal distribution with parameters

analytically related to the original ones [154]. Thus, the resulting precision matrix Θ_k, for $k = 1, \dots, K$ is

$$\Theta_k = \Sigma_k^{-1} = \left[\begin{array}{c|c} \Theta_k[M_i^k] & \Theta_k[M_i^k O_i^k] \\ \hline \Theta_k[O_i^k M_i^k] & \Theta_k[O_i^k] \end{array}\right], \tag{58}$$

and mean vectors are $\mu_k = (\mu_k[M_i^k], \mu_k[O_i^k])$.

The inference problem can then be defined as MLE. Note that, the likelihood has the same form for each k, and it is defined exploiting information on the observed data only [234, 266]. Nonetheless, the inference aims at learning all parts of the precision matrices Θ_k. The log-likelihood writes out as follows:

$$\ell(X_k[O^k]|\Theta_k) = \frac{1}{2}\sum_{i=1}^{n_k}\left(\log\det(\Theta_{kO_i^k}^{-1}) + (X_{kO_i^k} - \mu_{kO_i^k}^\top)(\Theta_{kO_i^k}^{-1})^{-1}(X_{kO_i^k} - \mu_{kO_i^k})\right). \tag{59}$$

Finally, given the likelihood, the kernel κ and a behaviour Ψ the functional to optimise to perform inference takes the form

$$\underset{\boldsymbol{\Theta}\in\mathcal{S}_+^d}{\text{minimize}} \sum_{k=1}^{K}\left[-\ \ell(X_k[O^k]|\Theta_k) + \alpha\|\Theta_k\|_{\text{od},1}\right] + P_{\Psi,\kappa}(\boldsymbol{\Theta}) \tag{60}$$

where $\|\cdot\|_{od,1}$ is the off-diagonal ℓ_1-norm, which promotes sparsity in the precision matrices and whose strength is regulated by α and the constraint $\Theta_k \in \mathcal{S}_+^d$ restricts the search space to the cone of positive definite matrices.

6.1.5 *Latent data specialization*

The optimisation problem presented in Equation (60) requires further considerations when in presence of latent data. Indeed, when introducing latent variables while the model remains the same we naturally introduce a further hyper-parameter. Consider the set M of missing values, in the case of latent variables such set is assumed to be the same across all k multiple graphs and across all n_k samples. Nonetheless, no prior information on the number of latent variables is naturally provided with the data. The number of latent variables is exactly the cardinality of the set$|M| = r$, and thus, by imposing a certain set M we are imposing a certain number of latent variables. Such hyper-parameter can be selected via model selection strategies, but it can be shown that results are not particularly subsceptible to this choice, i.e., we reach similar performances even if the value r is not precisely the same of the true underlying system [266].

The proposed model allows us to infer the relationships among latent variables and between latent and observed variables, thus obtaining an explicit information of them differently from what was done in [242].

Algorithm 4 EM algorithm

Inputs: Ψ consistency function, κ temporal dependencies, $\boldsymbol{X}$ samples,
α sparsity hyper-parameters
for $\iota = 1, \dots,$ **do**
  for $k = 1, \dots K$ **do**
    $c_{ki} = \mu_k[M_i^k] + K_k[M_i^k]^{-1}\Theta_k[M_i^k O_i^k](X_k[:, O_i^k] - \mu_t[O_i])$
    $\mathbb{E}[X_k[iv]|X_k[O_i^k], \mu_t^{\iota-1}\Theta_k^{\iota-1}] = \begin{cases} X_k[iv] & \text{if } v \in O_i^k \\ c_k[iv] & \text{if } v \in M_i^k \end{cases}$
    $\mathbb{E}[X_k[iv]X_k[iv']|X_k[O_i^k], \mu_t^{\iota-1}\Theta_k^{\iota-1}] = \begin{cases} X_k[iv]X_k[iv'] & \text{if } v, v' \in O_i^k \\ X_k[iv]c_k[iv'] & \text{if } v \in O_i^k \\ (\Theta_k[M_i^k]^{-1})_{vv'} + c_k[iv]c_k[iv'] & \text{otherwise} \end{cases}$
    $C_k^\iota[vv'] = \sum_{i=1}^{n_k} \mathbb{E}[X_k[iv]X_k[iv']|X_k[O_i^k], \mu_k^{\iota-1}\Theta_k^{\iota-1}]$
  end for
  for $k = 1, \dots K$ **do**
    $\mu_k^\iota = \frac{1}{n_k}(\sum_{i=1}^{n_k} X_k[i1], \dots, \sum_{i=1}^{n_k} X_k[id])$
  end for
  $\boldsymbol{\Theta}^\iota = \underset{\boldsymbol{\Theta} \succ 0}{\operatorname{argmin}} \sum_{k=1}^{K}[-n_k \ell_{GGM}(\Theta_k|C_k^\iota) + \alpha\|\Theta_k\|] + P_{\Psi,\kappa}(\boldsymbol{\Theta})$
end for

6.1.5.1 *Optimisation*

Problem (60) is non-convex because of the unknown values of the input data matrices $\boldsymbol{X}$. In the ideal case of complete input data, the estimation of the two sets of parameters $\boldsymbol{\mu}$, $\boldsymbol{\Theta}$ would be straightforward using optimisation methods as the Alternating Direction Methods of Multiplier (ADMM) [45]. Nonetheless, the lack of knowledge on input data requires a specific optimization method. We recur to the Expectation Maximization (EM) algorithm, an alternating procedure that has guarantees of reaching a local minimum of functional (60). The procedure consists of two steps the E-step and the M-step, both repeated at each iteration. In the former we compute the expectation on the missing values (thus estimating the complete $\boldsymbol{X}$) while in the latter we substitute the estimated complete data in the functional and we maximise it to retrieve distribution parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Theta}$. The EM algorithm for the complete data case is described in Algorithm 4. Such algorithm may get stuck in local optima, for this reason we may require multiple initialisations in order to detect the final best reliable network.

The E-step can be performed separately for each k because of the linearity of the expectation operator. In order to easily compute the expectation we express the likelihood in Equation (59) in terms of sufficient statistics of the Normal distribution. For each k we have two sufficient statistics: the sample mean μ_k^C and the sample covariance matrix $C_k = \frac{1}{n_k} X_k^\top X_k$. The E-steps consists of the estimation of the expectation of the two sufficient statistics μ_k^C and C_k, this can be done by observing that, thanks to the factorization property of the normal distribution, for each sample i the missing values indexed by M_i^k are again distributed according to a multivariate normal distribution. Thus, the expectation of the missing values can be computed as the mean of such distribution. By substituting the previous estimates for Θ_k and μ_k at iteration t we can compute

the expectation as in line 6 of Algorithm 4 Given the estimates of the E-step we can now reason in terms of complete data, and the maximisation problem corresponds to the model proposed in [241] and has the form in line 13 of Algorithm 4.

We call the implementation of this EM algorithm *Missing Multiple Graphical Lasso* (MMGL$_\kappa$), we typically omit the subscript κ if it is clear from the context. In general, it can either model the temporal graphical lasso or the joint graphical lasso by substituting kernels in Equations (55) and (56). Note that in [241] they proposed different kernels also for the modelling of complex temporal patterns. We do not mention them as it is not the aim of our work but we can substitute also these kernels to obtain more complex temporal patterns (*e.g.* seasonality).

6.1.6 *Experimental validation*

We assessed the performance of the proposed method in different scenarios compared with the state-of-the-art methods. Due to space restrictions, we refer the reader to the publicly available open source Python library `regain`[1] for complete details on the experiments. We designed three synthetic experiments:

- **Exp-PT**: we considered a temporal evolving graphical model (kernel of Equation (56)) in presence of partial data. We compared the performance with the time-varying graphical lasso (TGL) [112] coupled with pre-processing techniques (imputing and complete data strategies) and the missing graphical lasso (MGL) [234] that copes with partial data, but does not consider time (*i.e.*, we fit a model for each time point).
- **Exp-PC**: we considered a multi-class graphical model (kernel in Equation (55)) and we assessed performances of the state-of-the art method the Joint Graphical Lasso (JGL) [74] coupled with imputing and complete data.
- **Exp-LT** we consider a latent temporal evolving graphical model (kernel specified in Equation (56)) and we compared the performance with the state-of-the-art method for latent time-varying graphical inference (LTGL) [242] and a static version of an EM approach, LVGLASSO, that deals with latent variables [266].

For all these experiments we performed the analysis at five increasing values for the of observed variables $|O|$ in the interval $[10, 100]$. For each of these values we performed the experiments at three different percentages of missing data $5, 10, 20\%$ for both partial and latent cases. We fixed the number of samples $N_k = 100$ in order to consider an increasing ill-posedness as the number of observed variables increases. We fixed the number of classes/times $K = 10$, so that the total number of unknowns is given by the formula $K * \frac{|O|*(|O|-1)}{2}$.

1 https://github.com/veronicatozzo/regain

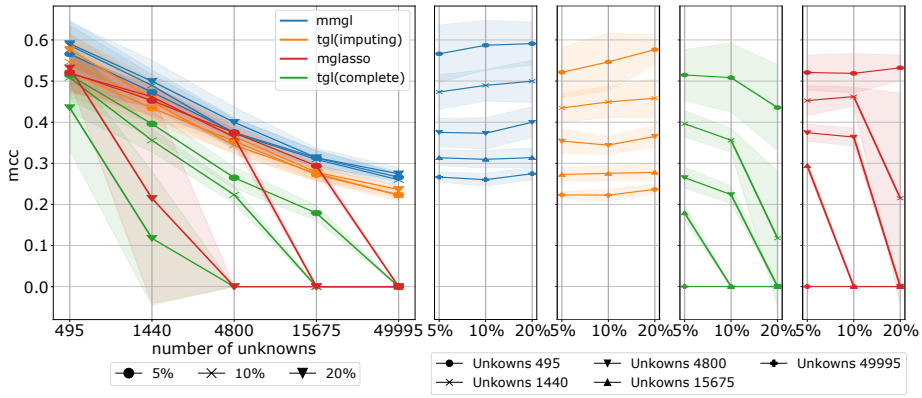


Figure 34: MCC of the results obtained for MMGL, TGL(complete), TGL(inputing) and MGL. Left panel all results, right panel mean results per percentage of partial data.

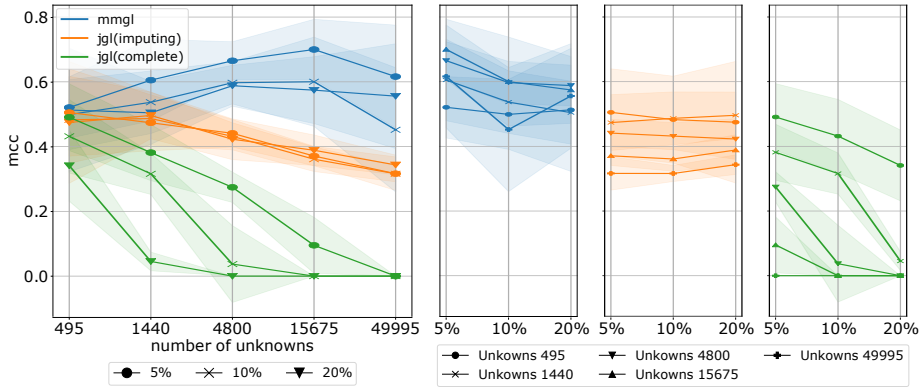


Figure 35: MCC of the results obtained for MMGL, JGL(complete), JGL(inputing). Left panel all results, right panel mean results per percentage of partial data.

For each of these possibilities we repeated the experiments 10 times to have mean results.

Given the large number of total experiments and hyper-parameters for each considered method performing ad-hoc cross-validation, even bayesian optimization method [198] would have required too much computational time [243]. Thus, we fixed some of them to the value provided by theoretical bounds, while we fixed the others to reasonable arbitrary values. The hyper-parameters are as follows. All the models have the sparsity enforcing parameter α, MMGL has κ which, if we use either Equation (56) or (55), it coincides with the choice of a parameter β similarly to TGL [112], JGL [74] and LTGL [242]. In the specific case of latent variables we also need to fix the number of latent variables r, similarly to LVGLASSO [266]. Lastly, LTGL has two further hyper-parameters τ and η that guide the latent variables behaviour in time. We fixed $\alpha = \frac{\log |O|}{N_k}$ [210] and $\tau = 2\sqrt{|O|/N_k}$ [55] according to theoretical results, while we fixed $\beta = 1$ and r to the real number of latent variables in synthetic data. While one may object that fixing r is not fair w.r.t. the choice of $\tau = 1$ for LTGL it can be shown that the choice of r does not impact heavily the performance of our algorithm and thus we opted to fix it to the true value. Such behaviour is similarly reported in [266]. Simulation results can be found in the github repository.

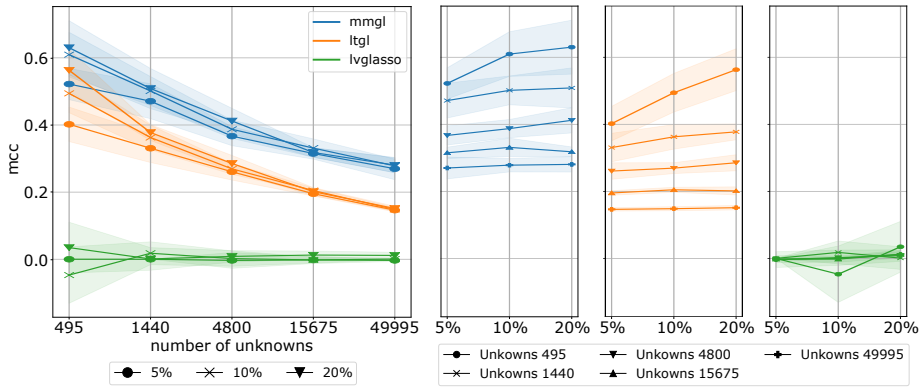


Figure 36: MCC of the results obtained for MMGL, LVGLASSO and LTGL. Left panel all results, right panel mean results per percentage of partial data.

All hyper-parameters were fixed to the same values for all the models under consideration.
We present the results in terms of Matthews Correlation Coefficient (MCC) [168] where we considered as true positive the number of edges correctly identified, true negative the number of edges that were correctly inferred as absent, false positive is the number of edges that the algorithm identifies as existing but are not and false negative are those edges that are not inferred by the algorithm but exist.

6.1.6.1 *Results*

Results are presented in Figure 34, 35 and 36 for **Exp-PT**, **Exp-PC** and **Exp-LT** respectively. It can be seen that MMGL is the one with the highest MCC across all types of experiments while being stable as the percentage of partial or latent data increases (right panel of all three figures). The only state-of-the-art method that provides closely related performance is TGL with imputing as pre-processing strategy. When using complete data as pre-processing strategies, as well as when exploiting static method, the percentage of missing data matters on the final result. Indeed, regularization for multiple join inference allows us to exploit information on other classes/time to improve the inference as it can be seen also for TGL and JGL with imputing. In Figure 34 and 36, we observed that 20% missing data result in better performances, while not having a theoretical explanation for this behaviour we argue that it might simply be due to the data generation process. We performed also a scalability assessment in terms of convergence time. Results are presented in Figure 37, where we can observed that MMGL is the one that requires the highest time to converge in the **Exp-PT** and **Exp-LT**, this is expected as MMGL solves a non-convex problems that also estimates the means and precision matrices in time. Instead, in **Exp-PT** we can observe that it is faster than the available state-of-the-art implementation of JGL.

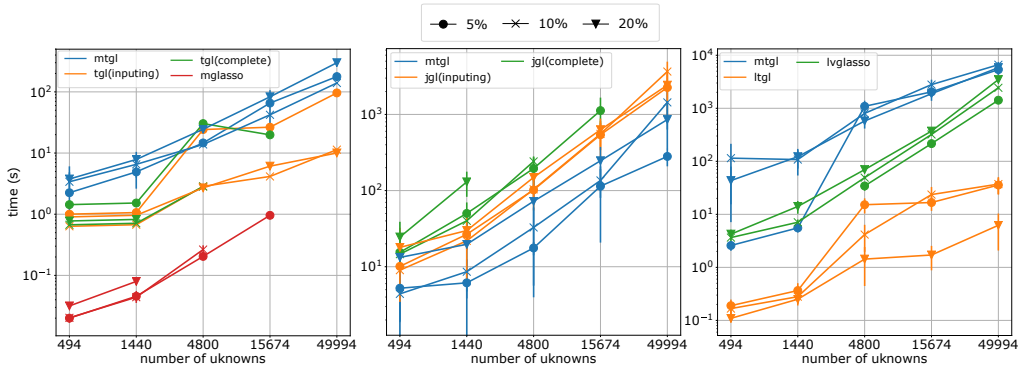


Figure 37: Scalability results for the experiments **Exp-PT** (left panel), **Exp-PC** (central panel) and **Exp-LT** (right panel).

6.1.6.2 *Case study: automobile safety*

We performed a last experiment on a real-world dataset that consists of a multi-class dataset of automobile characteristics[2] where class 0 corresponds to set of safest cars and class 5 to the most risky sett. The dataset contains $\approx$ 26% of missing values across all classes, which are also unbalances as we have 3 samples for class 0, 22 for class 1, 67 for class 2, 54 for class 3, 32 for class 4, and 27 for class 5. The hyper-parameters were selected through likelihood-based cross-validation in the following ranges $\alpha = [10^{-2}, 10^{2}]$ and $\beta = [10^{-5}, 10^{1}]$ and the resulting hyper-parameters are $\alpha = 10$ and $\beta = 10^{-4}$. We analysed the dataset with MMGL and JGL coupled with imputing, results are presented in Figure 38. The first thing that we can notice is that there are difference in the edges inferred in class 0 or 5 with the two methods as well as there are differences in the graph inferred with MMGL from the two classes but JGL tends to infer the same graph throughout all the classes. We do not discuss the possible insights obtained by this analysis as they are not the scope of this paper, but we want to remark that by exploiting MMGL we can obtain insights on the dataset that were impossible to obtain using JGL as it infers a network that is consistent across all five classes.

6.1.7 *Conclusions*

We presented a method that allows in a unique way to handle the inference of networks from data that may have different patterns of missing data as well as different complex inter-relationships as multi-class or temporal evolution. We showed on synthetic data that our method performs better than a variety of state-of-the-art methods thus providing a valuable alternative to both joint network inference method [74, 112, 242] as well as static inference methods that deal with missing data [234, 266]. MMGL optimization could be improved. Indeed, we could reduce time to converge by parallelising the K expectations computations. We do not provide such results but we expect the time to convergence to be reduced drastically. Moreover, in the case of latent variables we

2 `https://sci2s.ugr.es/keel/dataset.php?cod=90`

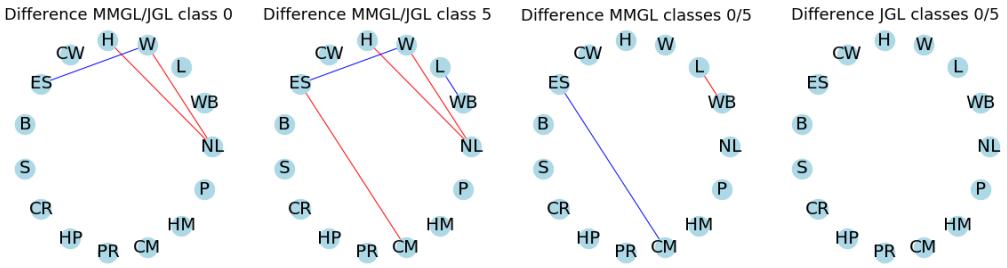


Figure 38: Comparison of the networks obtained with MMGL and JGL on the automobile dataset. A blue edge means that the edge was present in the first named set and not in the second. Red edge the opposite. The labels correspond to Normalized-losses (NL), Wheel-base (WB), Length (L), Width (W), Height (H), Curb-weight (CW), Engine-size (ES), Bore (B), Stroke (S), Compression-ratio (CR), Horsepower (HP), Peak-rpm (PR), City-mpg (CM), Highway-mpg (HM), Price (P).

can reason in terms of blocks when computing the expectation step and we can remove the estimate of the means as re-centering the data does not induce any bias. This approach is equivalent but would further reduce convergence time. Our derivations show that indeed handling latent data or partial data can be done in a unique way, and this show that the two different methods presented in [266] for latent data and [234] for partial data are, indeed, the same. As proposed in [234], this method could be exploit as a pre-processing step to perform imputing and thus preparatory to supervised machine learning tasks. We leave the validation of this approach to further work due to lack of space. Also, our method different from methods that marginalise out the latent variables effect as [54, 242] and, by allowing a direct estimate of the latent variables graph may be used for complex inference and data mining.

6.2 Non-stationarity in Time-varying Gaussian Graphical Models

Part of this section is present in the following publication: *Tozzo, Veronica, Ciech, Federico, Garbarino, Davide, & Verri, Alessandro (2021, August). Statistical Models Coupling Allows for Complex Local Multivariate Time Series Analysis. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining (pp. 1593-1603).*

The increased availability of multivariate time-series asks for the development of suitable methods able to holistically analyse them. To this aim, we propose a novel flexible method for data-mining, forecasting and causal patterns detection that leverages the coupling of Hidden Markov Models and Gaussian Graphical Models. Given a multivariate non-stationary time-series, the proposed method simultaneously clusters time points while understanding probabilistic relationships among variables. The clustering divides the time points into stationary sub-groups whose underlying distribution can be inferred through a graphical model. Such coupling can be further exploited to build a time-varying regression model which allows to both make predictions and obtain insights on the presence of causal patterns. We extensively validate the proposed approach on synthetic data showing that it has better performance than the state of the art on clustering, graphical models inference and prediction. Finally, to demonstrate the applicability of our approach in real-world scenarios, we exploit its characteristics to build a profitable investment portfolio. Results show that we are able to improve the state of the art, by going from a -%20 profit to a noticeable 80%.

6.2.1 *Introduction*

Local multivariate time series analysis involves a variety of tasks that aim at studying the characteristics of subsequent points in time, such as their short or long term similarities or effect on future behaviour.

Consider the example of stock trends in a financial system. Each stock has an associated value that changes in time. Such value could possibly depend on other stocks behaviour as well as external environmental causes. At each time point one may want to obtain various insights on such stock values. Here, we refer to the following scenarios: *(a)* understanding how stocks are related to each other (*multivariate correlation analysis*) [56] ; *(b)* detecting patterns of stock interactions repeated in time (*time points clustering*) [146, 213]; *(c)* predicting their value at the next time-point (*forecasting*) [57, 191, 272], and, lastly *(d)* understanding possible causal relationships among them (*non-stationary causality*) [204]. These tasks aim at locally characterizing every single observation in the time series under different perspectives. If applied on the same time series, these local characterizations can naturally enable a global understanding of the

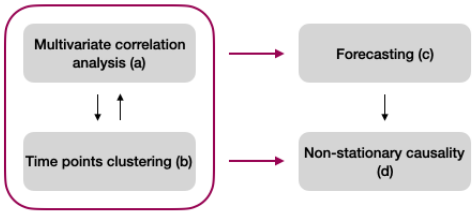


Figure 39: Schematic representation of the four tasks and how they impact each other in our framework. Multivariate correlation analysis (task *(a)*) and time point clustering (task *(b)*) are mutually supportive to one another, together (purple arrows) they allow for forecasting (task *(c)*) and the understanding of non-stationary causal patterns (task *(d)*). This last is also influenced by forecasting.

underlying system. Typically, these tasks are approached separately by restricting the problem to a constrained setting (*i.e.*, strong assumptions on data) with the intent of having the best possible performances on such domain [90, 100, 107, 130, 151, 189, 228, 249]. Nonetheless, we argue that these tasks are deeply connected to one another (Figure 39). Solving them globally may therefore result both in an increased accuracy per-task and a more in-depth insight on the system as a whole. On the latter, especially, the unified analysis makes the *a posteriori* global analysis easier to perform as the learned models are coherent with the initial assumptions and between each other by construction. The main contribution of this paper is a framework that solves tasks *a-d* globally, filling a missing gap in the literature. Our method synergically solves the four tasks by leveraging their dependencies as depicted in Figure 39. It achieves this goal with a simple coupling of two statistical models which allows to retain a single set of mild assumptions on the data. More formally, we present a statistical model, *Time Adaptive Gaussian Model* (TAGM), that combines Hidden Markov Models (HMMs) and Gaussian Graphical Models (GGMs). The main idea is presented in Figure 40. Given a time series $\{\mathbf{x}_1, \dots, \mathbf{x}_6\} \subset \mathbb{R}^5$, we assign to each time point a hidden state $\{z_1, \dots, z_6\}$ that, in the specific example, assumes $K = 2$ values k_1 or k_2. Suppose to observe the state k_1, all observations assigned to that state (*i.e.*, $\{x_1, x_2, x_5\}$) are then i.i.d. and assumed to be drawn from a multivariate Gaussian distribution $\mathcal{N}(\mu_1, \Theta_1^{-1})$. The associated distribution is inferred through a GGM from $\{x_1, x_2, x_5\}$. Specifically, a GGM univocally determines the underlying distribution through a graph that is encoded in the matrix Θ_1. Such matrix is called *precision matrix* and is the inverse of the covariance matrix. In the graph we can directly read pairwise conditional independencies among variables, *e.g.* $v_1 v_2 | v_3, v_4, v_5$ if and only if $\Theta_1(1,2) = 0$, or, equivalently, there is no edge between the two nodes in the corresponding graph. TAGM naturally solves tasks *(a)* and *(b)* (Section 6.2.2). Indeed, we can divide the time points into clusters by pairing each cluster to a hidden state. At the same time, multivariate correlation analysis is achieved through the inference of the corresponding underlying graph. This coupling is both advantageous and necessary. On the one hand, it is advantageous since knowledge on the variable dependencies can be leveraged to better assign each time

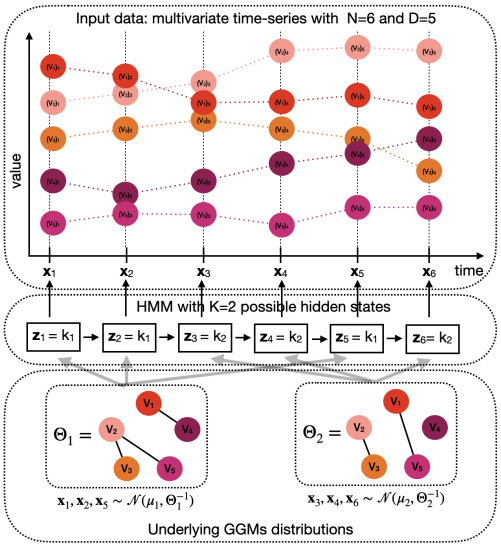


Figure 40: Schematic representation of the simultaneous instantiation of multivariate correlation analysis and time point clustering. Given a time series of multi-dimensional vectors, through an HMM, we associate each time point to a hidden state (*i.e.*, cluster). Given the K=2 hidden states, we infer two GGMs that model the underlying distributions. Note that, if we observe the Markov chain of the HMM, we can assume the time points associated to a specific state to be *i.i.d*.

point to a cluster. On the other hand, it is necessary as the inference of a GGM requires more than one observation. Thus, we can exploit the division into clusters to collect independent and equally distributed samples. TAGM can also be directly used for forecasting (task *(c)*, Section 6.2.3). At training time, we construct an augmented version of the input time series in which we concatenate each time point with the next one (Figure 44). We then apply TAGM on this augmented time series to infer the relative clusters and variable dependencies. At prediction time, we predict the next most probable state/cluster by exploiting the information on the previous inferred states in the HMM. The values of the unseen point are then estimated exploiting the GGM associated to that cluster. The same procedure can be used to understand causal patterns (task *(d)*, Section 6.2.4). Indeed, the learned statistical dependencies and sequentiality allow us to have insights on causal relationships. Note that, the ability of detecting more than one GGM allows for non-stationary causality patterns, which entails high flexibility.

We extensively evaluate TAGM on synthetic data sets and show how it leads to significant improvement compared to the state of the art in clustering, learning graph structures and prediction. Results are presented in Section 6.2.2.2 for tasks *(a)* and *(b)*, and Section 6.2.4.1 for task *(c)*. We also show the applicability of our method to real-world scenarios through a financial use case (Section 6.2.5). In particular, we exploit it to construct a profitable investment portfolio [273]. The standard approach used in finance is to fix a temporal window on which perform correlation analysis [203]. Such approach does not account for underlying changes and thus assumes all time points in that win-

dow to be stationary. Differently, our method removes such assumption while also exploiting the interpretability of GGMs to understand the dependencies among stocks. This is fundamental, as simple correlation analysis often carries spurious information that can lead to poor decision making. We show that TAGM overcomes the state-of-the-art approach for building a portfolio in terms of profit and loss variations, obtaining a 80% profit compared to the -%20 of standard methods.

6.2.2 *The Time Adaptive Gaussian Model*

TAGM is based on the assumption that the system under analysis is non-stationary, *i.e.*, the underlying distribution of variables at each time point may change in time. One of the possible methods to analyse sequential non-stationary data is a Hidden Markov Model (HMM) [28]. HMMs assume that the series of observations is generated by a certain number of (hidden) internal states connected through a Markov chain of latent variables. If we consider Figure 40, the latent variables are modelled as $\mathbf{z}_1, \ldots, \mathbf{z}_6$. Such variables may assume possible $K = 2$ different states, each of these gets associated to a possibly different distribution. The family of such distributions depends on the type of data in analysis, for simplicity, we consider continuous data and we assume underlying multivariate Gaussian distributions. Nonetheless, adopting the same idea for other distribution assumptions is straightforward from the model definition. Differently from standard approaches, instead of directly estimating the empirical means and covariances of such distributions from observations, we associate a graphical model where the lack of an edge explicitly encodes the conditional independencies between a pair of variables. We exploit Gaussian Graphical Models (GGMs) [147] where the precision matrices (Θ_1 and Θ_2) are the inverse of the covariance matrices and they can be interpreted as the adjacency matrices of a graph. This switch of perspective still allows us to estimate a multivariate Gaussian distribution, while allowing for directly imposing sparsity on the adjacency matrices [100]. Sparsity is fundamental as: it provides a higher stability to noise in presence of fewer samples; it grants higher interpretability as it allows for identifying the most relevant dependencies (*i.e.* the edges) while removing spurious correlations that would be captured by the empirical covariance matrix; and, it allows us to extend this method to perform prediction as well as understand causality patterns (more details about this will be provided in Section 6.2.3 and **??**).

6.2.2.1 *The model*

Consider N sequential (temporal) observations, $\mathbf{x}_n \in \mathbb{R}^D$ for $n = 1, \ldots, N$ on D variables. We assume that such observations follow a non-stationary process, and thus are generated by possibly more than one underlying distribution. The number of such distributions could be ideally infinite [32], but, for the sake of simplicity, we here assume them to be fixed to K. In order to map each observation $\mathbf{x}_n$ to one distribution, HMM pairs them to a hidden (latent)

variable $\mathbf{z}_n \in \{0,1\}^K$, that has only one non-zero component at position k if the n-th observation is associated to the k-th distribution. In the rest of the paper, we will use the notation $z_{n,k}$ to indicate the k-th positional value of the vector $\mathbf{z}_n$.

The sequence of latent variables follows a Markov chain process, meaning that $z_{n+1}z_{n-1}|z_n$. As a consequence, if we condition on the latent variables, the observations $\mathbf{x}_n$ and $\mathbf{x}_{n+1}$ become independent thus allowing to freely use them to infer the corresponding underlying distributions (see Figure 40).

The joint distribution on observed $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)$ and latent variables $\mathbf{Z} = (\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N)$ is given by

$$p(X, Z|\pi, A, \phi) = p(z_1|\pi)\Big[\prod_{n=2}^{N} p(z_n|z_{n-1}, A)\Big]\prod_{n=1}^{N} p(x_n|z_n, \phi), \tag{61}$$

where [28]:

- the probability $p(z_1|\pi) = \prod_{k=1}^{K} \pi_k^{z_{1,k}}$, with $\sum_k \pi_k = 1$, is the initial latent node z_1 probability, which differs from the other states as there is no parent node;
- the probability $p(z_n|z_{n-1,A}) = \prod_{k=1}^{K}\prod_{j=1}^{K} A_{j,k}^{z_{n-1,j}z_{n,k}}$ is the *transition probability* of moving from one state to the other. Here, $A \in [0,1]^{K\times K}$ is the *transition matrix* that we assume to be constant in time and it is defined as
$$A_{j,k} = p(z_{n,k} = 1|z_{n-1,j} = 1)$$
with $0 \leq A_{j,k} \leq 1$ and $\sum_k A_{j,k} = 1$.
- the probabilities $p(x_n|z_n, \phi) = \prod_{k=1}^{K} p(x_n|\phi_k)^{z_{n,k}}$, are the *emission probabilities* where $\phi = \{\phi_1, \ldots, \phi_K\}$ is a set of K different parameters governing the distributions, one for each of the possible K states.

Combining HHMs with GGMs is achieved by setting the emission probabilities to

$$p(x_n|z_n, \phi) = \prod_{k=1}^{K} \mathcal{N}(x_n|\mu_k, \Theta_k^{-1})^{z_{n,k}}$$

in such a way to have an explicit correspondence between the distribution of each state and a graph, modelled through the precision matrix Θ_k.

Moreover, we impose sparsity by coupling the emission probabilities with a Laplacian prior on the precision matrices. The posterior of the final model is defined as

$$\begin{aligned} p(\pi, A, \mu, \Theta|X, Z) \propto & p(z_1|\pi)\Big[\prod_{n=2}^{N} p(z_n|z_{n-1}, A)\Big] \\ & \prod_{n=1}^{N}\prod_{k=1}^{K} \mathcal{N}(x_n|\mu_k, \Theta_k^{-1})^{z_{n,k}} e^{-\frac{\lambda}{2}||\Theta_k||_{1,od}}. \end{aligned} \tag{62}$$

OPTIMIZATION The optimization of parameters $\theta = \{\pi, A, \bar{}, \Theta\}$ in functional (62) is performed through a Maximum A Posteriori approach. In particular, we employ the Baum's version of the *expectation maximization* (EM) algorithm [29]. In short, it consists in an alternating minimization procedure, in which the parameters $\pi, A, \bar{}$ are updated as in a standard HMM, while the inference of the $\boldsymbol{\Theta}$ reduces to solving a Graphical Lasso [100]

$$\Theta_k = \underset{\Theta \succ 0}{\operatorname{argmin}} \ \operatorname{tr}(\Theta \mathbf{S_k}) - \log\det(\Theta) + \lambda \|\Theta\|_{1,od} \tag{63}$$

where S_k is the empirical covariance matrix of the observations that are associated to the k-th hidden state, and if we denote such observations as X_k, it is defined as

$$\mathbf{S}_k = \frac{1}{\sum_{n=1}^{N} z_{nk}} \sum_{n=1}^{N} z_{nk}(\mathbf{x}_n - \hat{\boldsymbol{\mu}}_k)(\mathbf{x}_n - \hat{\boldsymbol{\mu}}_k)^{\mathrm{T}}, \hat{\boldsymbol{\mu}}_k = \frac{1}{\sum_{n=1}^{N} z_{nk}} \sum_{n=1}^{N} z_{nk}\mathbf{x}_n.$$

The objective value $\operatorname{tr}(\Theta S_k) - \log\det(\Theta)$ is the negative log likelihood of the multivariate normal distribution and $\|\cdot\|_{1,od}$ is the off-diagonal ℓ_1-norm that imposes sparsity on the precision matrix Θ without considering the diagonal elements. More details on the optimization of this model can be found in [66].

HIGHER ORDER AND ONLINE EXTENSIONS TAGM could benefit from two extensions to better handle real-world scenarios. The first extension is an online learning variation, *Incremental TAGM* or *IncTAGM*, that could be useful in presence of high-frequency data. Indeed, in real-world contexts where new observations arrive at a high rate (*e.g.* every second) one may want to be able to fine tune the model online in order to consider such observations instantaneously instead of re-fitting on the complete time series. We derive such extension following the idea in [62]. In practice IncTAGM starts as a standard TAGM on the available data, as a new time point becomes available the set of parameters is updated accordingly and the new point gets assigned to a state. Such update can be achieved by a variation in the E step of the EM algorithm, more details in [66].

The second extension is a higher-order Markov chain version, *Memory TAGM* or *MemTAGM*. It is based on the idea that latent states could have higher order dependencies, which are not captured if we consider a Markov chain of order one. In short, the main idea is to allow the emission probability of x_n to depend not only on z_n but also from the previous $m \in \mathbb{Z}^+$ sequence of states $p(x_n|\{x_\ell\}_{\ell<n}, \{z_\ell\}_{\ell\leq n}) = p(x_n|\{z_\ell\}_{\ell=n-(m-1)}^{n})$. Each observation is conditionally independent of the previous ones and of the state sequence history, given the current and the preceding $m-1$ states. For the derivation of the related optimization algorithm we followed [110]. The main drawback is a much higher computational time as the transition matrix and the corresponding initial state dimensions increase, more details in [66].

6.2.2.2 *Experimental assessment*

In order to evaluate the performance of TAGM we devised three sets of synthetic experiments. We generated data fixing the number of states K; for each

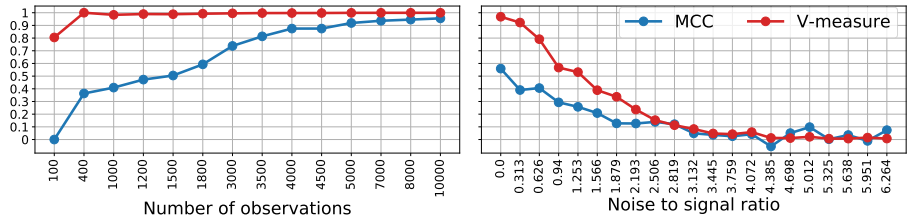


Figure 41: Asymptotic behaviour of TAGM on data. In the left panel we study the performance of the method as the number of observations increases, while on the right panel we fix the number of observations and we study the behaviour as the noise to signal ratio increases.

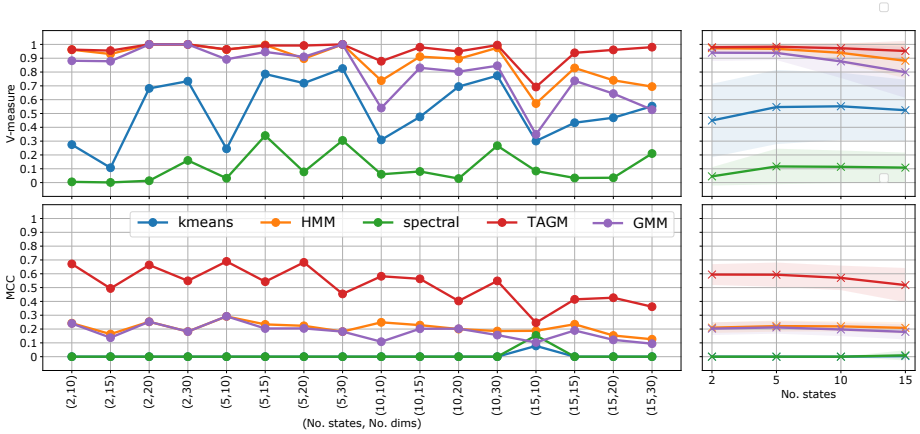


Figure 42: Comparison of TAGM with state-of-the-art methods in terms of V-measure and MCC. On the left panel we have the behaviour for different states and increasing dimensions, on the right panel the mean behaviour for the number of states.

state $k = 1, \ldots, K$, the mean $\mu_k \in \mathbb{R}^D$ is assumed to be drawn from a multivariate standard normal distribution $\mathcal{N}(0, \mathcal{I})$, while the precision matrix $\Theta_k \in \mathcal{S}^{D \times D}$ is generated as a sparse semi-positive definite matrix. The sequence of hidden states is generated by sampling from a transition matrix $A \in [0,1]^{K \times K}$ where each row is sampled from a Dirichlet distribution (see Appendix for more details). Given the sequence of states and the related μs and Θs, we draw a sequence of N samples in D dimensions. TAGM has two hyper-parameters, the number of hidden states K and the sparsity penalty λ in the Graphical Lasso (see Equation (63)). We cross-validated such parameters using the Bayesian Information Criterion (BIC) [224] (see Appendix 8.2). Note that the problem is non-convex, thus different initialization may lead to different local minima. To handle this issue, we performed multiple initializations and we select the result with highest likelihood (possible ways for initializing the model are described in Appendix 8.3). Results are presented in terms of V-measure for clustering performance [217], and Matthews correlation coefficient (MCC) for network inference performance [168] where we binarise the inferred precision matrix in such a way that 0 corresponds to a missing edge and 1 indicates an identified edge, see Appendix 8.4 for details. The optimization algorithm

and data generation pipelines for the experiments are implemented within an open-source Python framework that contains the real-world dataset as well [3].

STUDYING ASYMPTOTIC BEHAVIOUR As a first assessment we characterized the model in terms of number of observations needed to learn the data structure and its sensitiveness to external noise.On both experiments we fixed $D = 10$ variables and $K = 5$ states, for the first experiment we incremented the number of observations N until perfect inference is reached, while in the second experiment we fixed $N = 2000$ and we let the noise to signal ratio increments until the performances are equal to chance. The results are shown in Figure 41. As can be seen from the left panel of Figure 41, our model is able to converge to the real cluster labels after 400 observations while to infer the real graphs is necessary to have 10000 observations. On the right panel of Figure 41 we observe, as expected, that TAGM performance decreases as the noise standard deviation increases. We can also note that model performance remains good up to the point where noise to signal ratio is equal to one. Beyond that point the model is not able to distinguish the signal from the noise and therefore it reaches the performance of a random model. Given these results, for the following experiments we set the number of observations to $N = 2000$ and the noise to signal ratio to 1.

CLUSTERING AND NETWORK INFERENCE PERFORMANCE Here, we wanted to assess the ability of TAGM to infer the correct states of the system and the related GGMs. We generated synthetic datasets allowing for both the number of states K and the number of dimensions D to vary in the sets $\{2, 5, 10, 15\}$ and $\{10, 15, 20, 30\}$ respectively and we fixed $N = 2000$. We compared TAGM with state-of-the-art methods, in particular HMM [28], Gaussian Mixture Models (GMM) [90], spectral clustering [189] and K-Means [163]. Of these methods the only one that directly provides an estimate of the underlying distribution is HMM, note that it provides the empirical covariance matrix that we need to invert to be able to compare it with the precision matrix. For the other methods, we first infer the clusters and then on the samples belonging to each cluster we perform Graphical Lasso separately. Results are shown in Figure 42. On the left we show the point per point behaviour of the methods as both the number of states and the number of dimensions vary, while on the right we show the mean behaviour across different dimensions for the different number of states. It is evident that TAGM is the one that performs best in both V-measure and MCC and that HMM and GMM have close performance in clustering but have less ability in detecting the true graph.

HIGHER ORDER AND ON-LINE EXTENSION We finally performed two experiments to compare TAGM to the online and the higher order extensions. To compare TAGM and IncTAGM, we generated a synthetic dataset with $K = 5$ states, $D = 10$ dimensions and $N = 2000$ observations. We wanted to assess the behaviour of IncTAGM with respect to TAGM as the percentage of initial

3 https://github.com/veronicatozzo/regain/tree/HMM

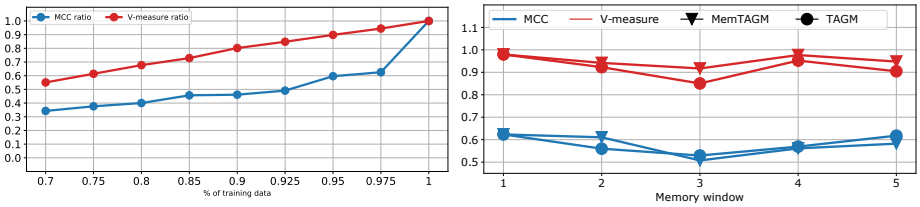


Figure 43: Comparison of TAGM with its extensions in terms of V-measure and MCC. On the left hand panel we drew the ratio V-measure(TAGM)/V-measure(IncTAGM) and the ratio MCC(TAGM)/MCC(IncTAGM) as the percentage of training data increases. On the right hand panel we drew the V-measure and MCC of MemTAGM and TAGM as the memory of the hidden Markov process increases.

data given in input $\hat{N} = \%N$ to IncTAGM increases. The results are shown on the left panel of Figure 43, where we can see that IncTAGM has reasonable performances when the percentage is low and it asymptotically tends to the performance of TAGM as the percentages of input data reaches 100%. To compare TAGM and MemTAGM we generated a synthetic dataset with $K = 3$ states, $d = 10$ dimensions and $N = 2000$ observations, while letting the memory of the hidden Markov process vary in the set $\{1, \dots, 5\}$. In this way we are able to evaluate the behaviour of MemTAGM with respect to TAGM as the memory of the hidden Markov process increases. The results are shown on the right panel of Figure 43, where we can see that MemTAGM performance in terms of V-measure is slightly better than TAGM for every considered memory window. On the other hand, the MCC results are comparable. We want to point out that the time complexity of the step of the optimization algorithm that assigns each point to a state is $O(K^2N)$, and MemTAGM number of states has a number of possible states $\hat{K} = K^\nu$ (with ν being the memory window), thus having complexity $O(K^{\nu 2}N)$. Therefore, the slight improvement in performances that we see in Figure 43 does not justify the need for a such increased complexity and therefore higher computational time.

6.2.3 *Making predictions*

Being able to predict future time points may be useful in applied contexts in which, for example, we seek to make decision based on unseen data. Here, we aim at performing a multi-output regression where given the values at time point n, $\mathbf{x}_n \in \mathbb{R}^D$, we want to predict the values at time point $n + 1$, denoted as $\mathbf{y}_n \in \mathbb{R}^D$.

If we are provided N observations in D variables, in order to exploit TAGM to predict the observation at $N + 1$, we first need to augment the time series. In practice for each time point $n = 1, \dots, N - 1$ we stack $\mathbf{x}_n$ and $\mathbf{y}_n = \mathbf{x}_{n+1}$ in a new vector $\hat{\mathbf{x}}_n = [\mathbf{x}_n, \mathbf{y}_n] \in \mathbb{R}^{2D}$ (see Figure 44).

If we now apply TAGM on the newly built time series, we obtain, for each time point its state and related underlying GGMs.

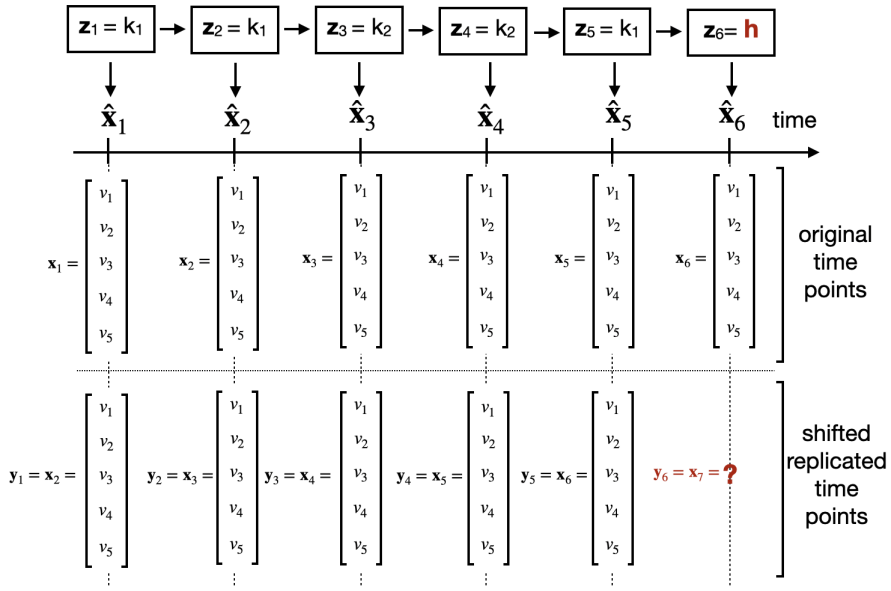


Figure 44: Schematic example of the construction of the augmented time series given in input to TAGM to perform predictions.

Suppose now that, at time n, we inferred a hidden state k with the related precision matrix $\Theta^k \in \mathbb{R}^{2D \times 2D}$. Such matrix can be divided in blocks as

$$\Theta^k = \begin{pmatrix} \Theta^k_{xx} & \Theta^k_{xy} \\ \Theta^{k\top}_{xy} & \Theta^k_{yy} \end{pmatrix}$$

where Θ^k_{xx} indicates the sub-matrix that encodes the conditional independencies of the vector $\mathbf{x}_n$, Θ^k_{yy} denotes the sub-matrix of conditional independencies of the vector $\mathbf{y}_n$ and the block Θ^k_{xy} encodes the conditional independencies among $\mathbf{x}_n$ and $\mathbf{y}_n$. Similarly, we can divide the inferred means vector as $\boldsymbol{\mu}^k = [\boldsymbol{\mu}^k_x, \boldsymbol{\mu}^k_y]$.

We can now observe that each y_n is normally distributed, indeed

$$p(y_n | x_n, z_{n,k} = 1) = \mathcal{N}(\bar{\boldsymbol{\mu}}, \bar{\Theta}^{-1}) \tag{64}$$

where

$$\bar{\boldsymbol{\mu}} = \mu^k_y + \Theta^{k\ T}_{xy} (\Theta^k_{xx})^{-1} (x_n - \mu^k_x), \tag{65}$$

and,

$$\bar{\Theta} = \Theta^k_{yy} - (\Theta^k_{xy})^\top (\Theta^k_{xx})^{-1} \Theta^k_{xy}. \tag{66}$$

Then, given a time-series of length N, and an inferred TAGM model on the augmented time series on the first $1, \dots, N-1$ time points, we aim at estimating the unknown values of y_N, given the observed x_N as $y_N = f(x_N)$. It is trivial to observe that the minimization of the expected squared prediction

error $f(x_N) = \mathbb{E}[y_N | x = x_N]$ corresponds to Equation (65). Thus, we predict the value of y_N as

$$y_N = \mu_y^h + {\Theta_{xy}^h}^T (\Theta_{xx}^h)^{-1}(x_N - \mu_x^h) \tag{67}$$

which corresponds to a time-varying lasso linear regression [99]. This regression model assumes the knowledge of h, *i.e.*, the hidden state assigned at time point N. Such state cannot directly be inferred from data because we do not have the complete values for $\hat{x}_N$, but it can be estimated propagating the information from the Markov chain of the HMM. To this end we exploit the Viterbi method [94].

This approach not only allows us to estimate the values of y_N, it also provides information on the predicted underlying GGM whose precision matrix is obtained as in Equation (66).

Moreover, this approach is flexible to consider more than one previous time point for the prediction of y_N. Indeed, if we want to exploit information on a window of length w, it is sufficient to build an augmented time series where, for each $n = 1, \ldots, N-1$, we define $\hat{x}_n = [x_{n-w}, \ldots, x_n, y_n] \in \mathbb{R}^{Dw}$.

6.2.4 *Causality*

Granger causality test [107]. Multivariate Granger causality analysis aims at detecting those variables that across all time series are causal for other. Typically, this is achieved by fitting an autoregressive model on the time series. The main drawback of this approach is that it assumes that all the observations are *i.i.d.* and, therefore, that causal relations do not change in time. TAGM, on the other hand, provides more flexibility and interpretability in this matter as to each of the K state is associated a different causal pattern given by the inferred precision matrix Θ^k. By looking at Equation (67), we can observe that the regression coefficients are given by $W = {\Theta_{xy}^h}^T (\Theta_{xx}^h)^{-1}$. Thus, for each variable $y[j]$ for $j = 1, \ldots, D$ the features that are causal for it are given by the coefficients in the j-th column of W. The causality of this is simply implied by the sequentiality of the data. Under a different perspective, our predictive model (Equation (67)) can be seen as a solution of an ordinary differential equation that models mass-action kinetics as specified in [204], Equation (2). Thus, TAGM allows us to detect possibly K multivariate non-stationary causality patterns in any input time-series.

6.2.4.1 *Experimental assessment*

We evaluated the performance of TAGM for prediction on one synthetic experiment. Data are fixing $K = 2, 3, 4, 5$ and $N = 2000$. The generation method is as described in Section 6.2.2.2 and Appendix 8.1. The hyper-parameters are cross-validated with BIC (Appendix 8.2) and results are presented in terms of Mean Absolute Error (MAE) (Appendix 8.4). For the estimate of the next time point precision matrix we compared the performances of TAGM with the inverse of the empirical covariance matrix of the last 25, 50 and 100 days on a

Method	K=2	K=3	K=4	K=5
Emp Cov last 25 days	0.23 ± 0.03	0.14 ± 0.07	0.00 ± 0.09	0.04 ± 0.12
Emp Cov last 50 days	0.29 ± 0.02	0.17 ± 0.13	0.02 ± 0.06	0.09 ± 0.15
Emp Cov last 100 days	0.31 ± 0.05	0.12 ± 0.11	0.04 ± 0.11	0.05 ± 0.09
TAGM	$\mathbf{0.65 \pm 0.03}$	$\mathbf{0.56 \pm 0.09}$	$\mathbf{0.62 \pm 0.11}$	$\mathbf{0.67 \pm 0.12}$

Table 9: Performance in the prediction of the next precision matrix in terms of MCC.

Method	K=2	K=3	K=4	K=5
Lgb	1.17 ± 0.23	1.43 ± 0.46	1.95 ± 0.45	1.55 ± 0.63
LSTM	1.16 ± 0.26	1.42 ± 0.45	2.06 ± 0.51	1.50 ± 0.38
VAR	1.14 ± 0.24	1.43 ± 0.45	1.97 ± 0.45	1.37 ± 0.36
Kernel RBF	1.15 ± 0.24	1.43 ± 0.45	1.97 ± 0.49	1.47 ± 0.46
TAGM	$\mathbf{1.09 \pm 0.23}$	$\mathbf{1.40 \pm 0.46}$	$\mathbf{1.93 \pm 0.41}$	$\mathbf{1.35 \pm 0.34}$

Table 10: Performance in the prediction of the next time point values in terms of MAE (below table).

time series of dimension $D = 10$. The results are in Table 9 where we observe that TAGM greatly outperforms the prediction compared to the estimate of the covariance matrix. For the evaluation of the prediction of the specific values we compared our model with Gradient Boosting (LGB) [151], Long-Short Term Memory Neural Network (LSTM) [130], Kernel regression with Gaussian assumption (Kernel RBF) [249] and vector autoregression (VAR) [228] on a time-series of $D = 5$ variables. The results are in Table 10 where we observe that TAGM has always a lower prediction error compared to all the other considered methods.

6.2.5 *Use Case: Stock Prices*

TAGM can be exploited to analyse stock market prices. In particular we consider the tasks of building an investment portfolio as well as forecasting of future stock values.

BUILDING AN INVESTMENT PORTFOLIO Ideally, a portfolio consists in set of stocks on which one invest. The best portfolio is one that provides the highest possible profit while maintaining a low fixed risk level. *Stock picking* (*i.e.*, the selection of the best stocks to put in the portfolio) is a hard task, indeed, even if you restrict to a given industrial sector, there are many factors that can cause underlying variations in the market. Moreover, stocks may be dependent on each other in a way that is often difficult to disentangle. The ability to detect and understand stock dependencies as well as changes in the market would

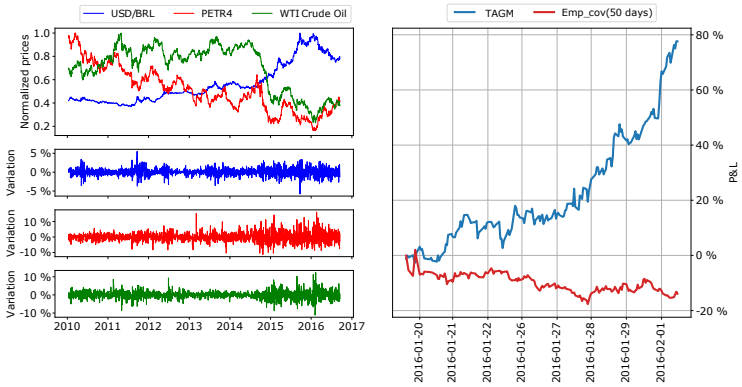
allow to perform the best *hedging* strategy. Hedging is the process of investing in contrary or opposite sectors in order to balance against a possible loss.

Nowadays, the study of stock dependencies is performed by fixing a temporal window and inferring the related empirical covariance matrix. Such method does not account for possible underlying changes in the distribution in that window, and, moreover, it requires to fix an arbitrary cut-off in the length of the analysed time series. TAGM, on the other hand, solves both problems as it could be applied as an exploratory step on the complete time series, while automatically detecting when the underlying distribution changes possibly due to environmental or political conditions. Simultaneously, it provides a cleaner view on the dependencies than the empirical covariance matrix as the GGMs graphs remove spurious dependencies among stocks.

We performed a small experiment by considering three securities (*i.e.* tradable financial assets): Petrobras (PETR4), WTI crude Oil front futures, and exchange from US dollar to Brazilian Real (USD/BRL). We considered the period from 12/01/2010 to 15/09/2016, corresponding to 1635 trading days which have many price swings (up and down) (see Figure 45 leftmost panel). We trained our model on the first 1470 days and we tested on the last 165 (from 20/01/2016 to 15/09/2016). Such securities have deep investment connections and the goal is to find a combination of weights (*i.e.*, amount of invested money) for each security which allows to earn a positive return in the long run and being backed from big losses in case of price oscillations (*i.e.* keeping a fixed risk). According to the Markowitz mean-variance portfolio optimization theory the two quantities of interests are: the expected value of returns and their covariance [41]. As these two quantities vary, the portfolio weights should vary accordingly.

To this end, each day starting from 20/01/2016, we fit TAGM on all the previous observations of the time series. The inferred GGM at the current date is used as weights for the securities. Note that, we adjust such weights if and only if there has been a change in the underlying distribution (*i.e.* the hidden state of the HMM) otherwise we keep the weights fixed to the previous day values. We compared the performance of our approach against the common state-of-the-art method of taking the last 50 days covariances. The performances are given in terms of profits and losses (P&L) and in the evaluation we suppose for simplicity that there are no trading fees. Results are shown in Figure 45 central panel where it is possible to see how TAGM greatly outperforms the empirical covariance approach going from a -20% to a 80% profit. This is due to the changes in trend of the WTI Crude Oil security, that are captured by TAGM but not by the mean empirical covariance strategy.

STOCK VALUES FORECASTING To test TAGM regression performance on real data we considered a period of 30 trading days from 4/08/2016 to 15/09/2016. We compare our results with the same state-of-the-art regression methods used in synthetic experiments. Figure 45 right panel shows that the performance of all the considered methods are very closed to each other, with TAGM slightly improving overall. This is due to the fact that past price values are not very



	MAE
TAGM	**0.459± 0.371**
Kernel RBF	0.462 ± 0.380
LSTM	0.461 ± 0.412
VAR	0.460 ± 0.386
Lgb	0.459 ± 0.396

Figure 45: TAGM enables significantly improvement in the construction of a financial portfolio (central panel) due to its ability to promptly detect changes in underlying dependencies among stocks (data showed on the left). TAGM shows better performance also in the prediction of next day stock prices (right panel).

informative to predict the next values since their short-run movements depend on the real time news and market sentiment. Therefore, as long as we do not introduce this information in the prediction it would be difficult to evaluate the prediction performance as all methods catch the same information (typically just noise).

6.2.6 *Related Work*

We position TAGM against various state-of-the-art approaches that perform clustering, temporal network inference, forecasting and causality analysis. All these approaches perform these tasks in isolation whereas TAGM can be used to simultaneously perform all four.

The combination of inferring a latent representation as clustering coupled with GGMs was presented in [90, 158], where the authors combined GGMs with Gaussian Mixture Models. Note that, such approach does not explicitly account for sequential data.

In the context of graph inference, TAGM can be seen as a generalisation of current state-of-the-art methods for temporal graphical models inference. Such methods typically rely on the assumption that the time points within a chunk, which size is arbitrarily chosen, are *i.i.d.* [97, 112]. Our approach relaxes such assumption making it more reliable for real-world analysis. We can find methods that perform simultaneous graph inference and clustering [113, 241]. Nonetheless, being based on the assumption of chunks they are impossible to compare directly with the presented model. All the previous methods have to rely on the imposition of norms to consider sequentiality. We avoid such imposition by relying on the Markov chain of the hidden states. In literature, we also find papers that look at single time points assuming local topological changes [56, 120, 184, 262] but they do not provide a way to directly perform clustering of the inferred networks.

For prediction tasks, we found in literature two examples that explicitly consider non-stationarity and sequentiality in a setting similar to ours. In [49] the authors use a vector autoregressive model (VAR) on time-dependent splines while in [184] they infer a dynamic graphical model and they predict the topology of the next time point. These last two methods, while allowing for next time point prediction, do not directly allow us to estimate the underlying precision matrix. Another interesting relationship of TAGM is with multivariate gaussian process regression [60], which makes explicit use of the conditional dependencies to estimate future time points. We want to point out that many methods that perform prediction on time series exist. We do not explicitly report them as their integration with GGMs is not obvious.

Lastly, differently from our setting, causality is often studied assuming stationarity of the time series, thus causal relationships are inferred as constant in time [72, 130, 151, 204, 228, 249]. In literature, we can find research directions that consider non-stationarity [124, 196, 231], but, to the best of our knowledge, the explicit use of dynamic graphical models to this aim is not present in literature.

6.2.7 *Conclusions and Future Directions*

We present a novel methodology to perform data-mining, forecasting and understanding causality patterns on multi-variate time-series. Our method combines HMMs and GGMs, providing a way to simultaneously cluster non-stationary time-series into stationary sub-groups and for each cluster detecting probability relationships among variables through graphical model inference. This simultaneous inference is suitable to be transformed into a time-varying regression model that allows to make predictions on non-stationary time-series. Moreover, the regression coefficients can be interpreted as causal patterns. Our method generalizes many state-of-the-art methods and provides a wide range of analysis type to be performed on time series. Both synthetic and real experiments show that it does indeed outperform state-of-the-art method for clustering, network inference and prediction.

There are many improvements that could be performed. One could add flexibility in the detection of each observation state by making the transition probabilities (the matrix A) time-dependent [133]. Also, using a non-parametric Bayesian approach would allow us to transform TAGM into an infinite-state model [32] thus removing the problem of identifying the most suitable hyper-parameter K (the number of states). Moreover, TAGM could benefit from convergence analysis and faster optimization techniques as it requires a high computational time when dealing with long time-series as the inference of the Markov chain cannot be easily parallelized. Two future interesting directions could be to relax the assumption of Gaussian distributed data and thus, by changing the emission probabilities to graphical models that allow for other type of distributions (*e.g.*, Poisson, Binomial, or others) [10, 120, 134, 261]. Moreover, if one is interested just in exploiting graphical models to study causality, we want to point out an interesting resemblance between Equation

(2) in [204] that models system kinetics and Equation (8) in [260] that define a pairwise graphical model on a general exponential family distribution. To conclude, the urge to dissect the underlying system observed through time series has led current research to deeply rely on graphical models. The approach we presented reinforces the general understanding that, indeed, graphical models are a powerful tool to study time series under a variety of different perspectives.

7

Conclusions

We conclude this thesis presenting a brief recap of the major contributions. The first part of the thesis focuses on the extension of state-of-the-art machine learning and deep learning methods to graph modeled data. We shed light on criticalities regarding the application of machine learning algorithms to relational data. In particular, we emphasize the paramount importance of learning suitable graph vector representations to properly incorporate complexity in adopted statistical models and leverage links information (Chapter 2 and 3). In Chapter 4 we propose four adaptations of machine learning algorithms to graph-modelled data, i.e.:

1. thanks to a degree-based representation of Web sub-networks (*i.e.* websites), we understand their topological evolution based on the generative principle: central or peripheral. Moreover, this representation allows us to question the overall ubiquitous presence of scale-free networks in real-world examples (Section 4.1);

2. we propose a novel approach based on network analysis to infer a surface-specific and time-varying score for professional tennis players and use it in addition to players' statistics of previous matches to represent tennis match data (Section 4.2);

3. relying on graphlet counts we represent preterm infants motion. The coupled use of this representation and a probabilistic model allows us to discover motion patterns characterizing different neurological conditions (Section 4.3);

4. we provide an extensive description of a future application for academic collaboration recommendations. In this regard, we propose a preliminary analysis of an academic collaboration heterogeneous network, *i.e.* the MaLGa center. (Section 4.4).

The second part of the thesis focuses on methodologies specific to the probabilistic network inference macro-area (Chapter 5). More specifically, we investigate inference methods of Markov Random Fields under the assumption of Gaussian distributed data. In this context, we deal with time-varying data and overcome two critical issues about missing data and non stationary time series:

1. in the former case, we propose a method that generalises state-of-the-art approaches and deals with two types of missing data: partial and latent (Section 6.1);

2. in the latter case, we propose the TAGM model which extends the state-of-the-art approaches to non stationary time-varying graphical models. TAGM relies on a clustering approach to identify stationary points in the multivariate time series (Section 6.2).

8

Appendix A

8.1 Synthetic dataset generation

We generated data through a Markov process which controls the probability to remain in the same state or to go from one state to another one.

The synthetic data generation comprehend the following steps:

1. we fix suitable values for the number of observations N, the number of states K and the number of multivariate time series D.

2. for every state $k = 1, \dots, K$, we allow for several combinations of distributions to generate the observations.

   a) The mean can be drawn in two ways: from a multivariate normal distribution$\mu_k \sim \mathcal{N}(0, \mathcal{I})$, where $\mathcal{I}$ is the identity matrix, or from an *uniform distribution* with $\mu_k \sim \mathcal{U}(a, b)$ and $a, b \in \mathbb{R}$ with $a < b$. If $a \ll b$ then the generated cluster are more likely to be separated between each other.

   b) The covariance matrix Σ_k can be set in three ways: fixing a certain maximum degree for each node d, we randomly selected its neighbours and put deterministically the weights of the edges to $0.98/d$ to ensure positive definiteness of the resulting precision matrix [171, 266]; from the tool **scikit-learn.datasets** which generates a random symmetric, positive-definite matrix; from the precision matrix stressing the links between nodes, starting from the identity matrix and putting randomly ones in the off-diagonal places respecting the symmetric matrix constraint. In this way we are generating precision matrix with either strong links between nodes or no links at all. This case is interesting because in this way the networks corresponding to each state k are very different between each others like the case with means very far away.

3. each row of the transition matrix A is generate from a Dirichlet distribution $\mathrm{Dir}(\alpha)$ where $\alpha \in \mathbb{R}_+^K$. In particular, to not have too quick transitions from one state to another we impose $\alpha_i = \kappa \cdot \alpha_j$ with $i \neq j$ where i is the

index of the row transition we are drawing and α_j all the other element of α different than α_i. κ is also known as the *force* constant, in the sense that the bigger κ is the more likely the state i is respect to the others;

4. finally, for each time point n, the state k is drawn from the transition matrix A then the data are drawn from the related normal distribution $X_n \sim \mathcal{N}(\mu_k, \Sigma_k)$.

8.2 Model selection

Our model has two hyper-parameters to cross-validate:

1. the number of finite states K;
2. the regularization parameter λ which regulates the sparsity of the precision matrix Θ_k;

To estimate these two hyper-parameters we employ cross validation (CV) with a Bayesian Information Criterion (BIC) score. To determine the number of hidden states we use the BIC approach [224] which has the form

$$\text{BIC}(m) = \ln p(X|m, \theta) - \frac{\nu}{2} \ln(n).$$

ν represents the number of free parameters and m the considered model. In our case the number of free parameters can be computed in the following:

- the probabilities π have dimension K with one constraint, so $\nu_\pi = K - 1$;
- the transition matrix A has dimension $K \times K$ but each row has a constraint, so $\nu_A = K(K-1)$;
- the means μ are K with dimension d without any constraint, so $\nu_\mu = KD$;
- the precision matrices Θ are K, one for each state, with dimension $d \times d$ but they have the constraint given by graphical lasso therefore $\nu_\Theta = \sum_{i \geq j} e_{i,j}$ where $e_{i,j} = 0$ if $\hat{\Theta}_{i,j} = 0$ and $e_{i,j} = 1$ otherwise. $\hat{\Theta}$ is the estimated precision matrix.

Putting all together the total number of free parameter ν is

$$\nu = \nu_\pi + \nu_A + \nu_\mu + \nu_\Theta = (K-1)(K+1) + KD + \sum_{k=1}^{K} \nu_{\Theta_k}.$$

To see that this CV combination of methods is suitable for the estimation of the hyper-parameters of our model we generated a multivariate time series with $D = 10$ and $K = 5$ and we cross-validate K and λ from the sets $K \in \{3, \ldots, 8\}$ and $\lambda \in [18, 25]$. We show in Figure 46 the results and as we can see it found the K from which we have generated the data.

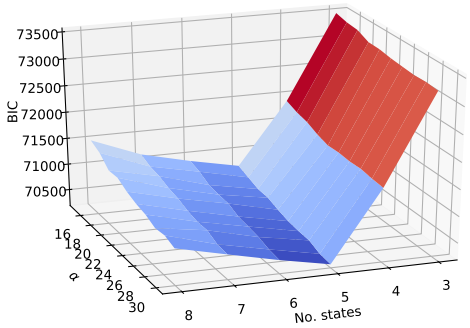


Figure 46: Cross validation

8.3 Initialization choices

The optimization algorithm requires an initialization of the initial parameters θ. Since the likelihood function we are considering is non-convex the parameters initialization is crucial to find the optimal solution. In particular,given the cluster number K we have to initialize four parameters: the transition matrix A and the initial probabilities π and the Gaussian distribution parameters Θ and μ. In our implementation we adopt the following initializations choices:

- **the transition matrix A and the initial probabilities π** can either be initialised with equal probabilities for each state $\frac{1}{K}$ or by randomly sampling from a uniform distribution $\mathcal{U}(0,1)$ or symmetric Dirichlet distribution $\mathrm{Dir}(1)$, with the constraints that each row has to sum to one;
- **the Gaussian distribution parameters Θ and μ** are initialized by computing an initial subdivision of the time points into clusters. To this aim with used K-means or Gaussian mixture model (GMM). Both GMM and K-means are non-convex, thus, depending on initialisation lead to different solutions as well. Given the dataset initial subdivision, we compute respectively the empirical covariances and means. Finally we run the graphical lasso to compute the corresponding precision matrix and we use that as initial parameters.

8.4 Evaluation metrics

We use a metric score for each of the following aspects:

1. **clustering performance**: we compare the clustering results in terms of V-measure [217] which returns a value $v \in [0,1]$ where $v = 0$ means that the cluster labels are assigned completely randomly while $v = 1$ means that there is a perfect match between the true labels and the one found by the models.
2. **network inference performance**: in order to evaluate the performances of the methods we need to identify a map between the true clusters and

the identified ones in order to compare the underlying graphs. Such map is obtained by taking the maximum per row of the contingency table of the true and predicted labels. We then consider the true and inferred graphs as binary classes (0 no edge identified, 1 edge identified) and we compute the Matthews correlation coefficient (MCC) [168]

$$\text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

which return a value in the interval$[-1, 1]$ where 0 corresponds to chance.

3. **forecasting performance**: we used as score the Mean Absolute Error (MAE) which measures the error between the true next point value and the predicted one. Since we are predicting d values for each future time point we compute the mean MAE across entries of the vector:

$$MAE = \frac{1}{n} \sum_{i=1}^{N} \left(\frac{1}{D} \sum_{d=1}^{D} |x_{nd} - \hat{x}_{nd}| \right)$$

.

Bibliography

[1] Hervé Abdi and Lynne J Williams. 'Principal component analysis.' In: *Wiley interdisciplinary reviews: computational statistics* 2.4 (2010), pp. 433–459.

[2] Fentaw Abegaz and Ernst Wit. 'Sparse time series chain graphical models for reconstructing genetic networks.' In: *Biostatistics* 14.3 (2013), pp. 586–599.

[3] Lars Adde, Jorunn L Helbostad, Alexander R Jensenius, Gunnar Taraldsen, Kristine H Grunewaldt, and Ragnhild Støen. 'Early prediction of cerebral palsy by computer-based video analysis of general movements: a feasibility study.' In: *Developmental Medicine & Child Neurology* 52.8 (2010), pp. 773–778.

[4] Monica Agrawal, Marinka Zitnik, and Jure Leskovec. 'Large-scale analysis of disease pathways in the human interactome.' In: *PACIFIC SYMPOSIUM ON BIOCOMPUTING 2018: Proceedings of the Pacific Symposium*. World Scientific. 2018, pp. 111–122.

[5] Amr Ahmed, Nino Shervashidze, Shravan Narayanamurthy, Vanja Josifovski, and Alexander J Smola. 'Distributed large-scale natural graph factorization.' In: *Proceedings of the 22nd international conference on World Wide Web*. 2013, pp. 37–48.

[6] David Ahmedt-Aristizabal, Simon Denman, Kien Nguyen, Sridha Sridharan, Sasha Dionisio, and Clinton Fookes. 'Understanding patients' behavior: Vision-based analysis of seizure disorders.' In: *IEEE journal of biomedical and health informatics* 23.6 (2019), pp. 2583–2591.

[7] Réka Albert. 'Network inference, analysis, and modeling in systems biology.' In: *The Plant Cell* 19.11 (2007), pp. 3327–3338.

[8] Réka Albert, Hawoong Jeong, and Albert-László Barabási. 'Diameter of the world-wide web.' In: *nature* 401.6749 (1999), pp. 130–131.

[9] Rubayyi Alghamdi and Khalid Alfalqi. 'A survey of topic modeling in text mining.' In: *Int. J. Adv. Comput. Sci. Appl.(IJACSA)* 6.1 (2015).

[10] Genevera I Allen and Zhandong Liu. 'A local poisson graphical model for inferring networks from sequencing data.' In: *IEEE transactions on nanobioscience* 12.3 (2013), pp. 189–198.

[11] Marilee C Allen. 'Neurodevelopmental outcomes of preterm infants.' In: *Current opinion in neurology* 21.2 (2008), pp. 123–128.

[12] Jeff Alstott and Dietmar Plenz Bullmore. 'powerlaw: a Python package for analysis of heavy-tailed distributions.' In: *PloS one* 9.1 (2014).

[13] Theodore W Anderson and Donald A Darling. 'A test of goodness of fit.' In: *Journal of the American statistical association* 49.268 (1954), pp. 765–769.

[14] David Aparício, Pedro Ribeiro, and Fernando Silva. 'Network comparison using directed graphlets.' In: *arXiv preprint arXiv:1511.01964* (2015).

[15] David Aparicio, Pedro Ribeiro, and Fernando Silva. 'Extending the applicability of graphlets to directed networks.' In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 14.6 (2016), pp. 1302–1315.

[16] Barry C Arnold. 'Pareto distribution.' In: *Wiley StatsRef: Statistics Reference Online* (2014), pp. 1–10.

[17] Niousha Attar and Sadegh Aliakbary. 'Classification of complex networks based on similarity of topological network features.' In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 27.9 (2017), p. 091102.

[18] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 'Neural machine translation by jointly learning to align and translate.' In: *arXiv preprint arXiv:1409.0473* (2014).

[19] Ting Bai, Youjie Zhang, Bin Wu, and Jian-Yun Nie. 'Temporal Graph Neural Networks for Social Recommendation.' In: *2020 IEEE International Conference on Big Data (Big Data)*. IEEE. 2020, pp. 898–903.

[20] Onureena Banerjee, Laurent El Ghaoui, and Alexandre d'Aspremont. 'Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data.' In: *Journal of Machine learning research* 9.Mar (2008), pp. 485–516.

[21] Albert-László Barabási and Réka Albert. 'Emergence of scaling in random networks.' In: *science* 286.5439 (1999), pp. 509–512.

[22] Albert-László Barabási, Réka Albert, and Hawoong Jeong. 'Scale-free characteristics of random networks: the topology of the world-wide web.' In: *Physica A: statistical mechanics and its applications* 281.1-4 (2000), pp. 69–77.

[23] Albert-László Barabási and Eric Bonabeau. 'Scale-free networks.' In: *Scientific american* 288.5 (2003), pp. 60–69.

[24] Albert-Laszlo Barabasi and Zoltan N Oltvai. 'Network biology: understanding the cell's functional organization.' In: *Nature reviews genetics* 5.2 (2004), p. 101.

[25] Albert-László Barabási et al. *Network science*. Cambridge university press, 2016.

[26] Tristan Barnett and Graham Pollard. 'How the tennis court surface affects player performance and injuries.' In: *Medicine and Science in Tennis* 12.1 (2007), pp. 34–37.

[27] Saeed Basirian and Alexander Jung. 'Random walk sampling for big data over networks.' In: *2017 International Conference on Sampling Theory and Applications (SampTA)*. IEEE. 2017, pp. 427–431.

[28] Leonard E Baum and Ted Petrie. 'Statistical inference for probabilistic functions of finite state Markov chains.' In: *The Annals of Mathematical Statistics* 37.6 (1966), pp. 1554–1563.

[29] Leonard E Baum, Ted Petrie, George Soules, and Norman Weiss. 'A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains.' In: *The Annals of Mathematical Statistics* 41.1 (1970), pp. 164–171.

[30] Martin Bax, Murray Goldstein, Peter Rosenbaum, Alan Leviton, Nigel Paneth, Bernard Dan, Bo Jacobsson, and Diane Damiano. 'Proposed definition and classification of cerebral palsy, April 2005.' In: *Developmental medicine and child neurology* 47.8 (2005), pp. 571–576.

[31] Nancy Bayley. *Bayley scales of infant and toddler development: administration manual*. Harcourt assessment, 2006.

[32] Matthew J Beal, Zoubin Ghahramani, and Carl E Rasmussen. 'The infinite hidden Markov model.' In: *Advances in neural information processing systems*. 2002, pp. 577–584.

[33] Punam Bedi and Chhavi Sharma. 'Community detection in social networks.' In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 6.3 (2016), pp. 115–135.

[34] Julian Besag. 'Spatial interaction and the statistical analysis of lattice systems.' In: *Journal of the Royal Statistical Society: Series B (Methodological)* 36.2 (1974), pp. 192–225.

[35] Smriti Bhagat, Graham Cormode, and S Muthukrishnan. 'Node classification in social networks.' In: *Social network data analytics*. Springer, 2011, pp. 115–148.

[36] Mansurul A Bhuiyan, Mahmudur Rahman, Mahmuda Rahman, and Mohammad Al Hasan. 'Guise: Uniform sampling of graphlets for large graph analysis.' In: *2012 IEEE 12th International Conference on Data Mining*. IEEE. 2012, pp. 91–100.

[37] E Bianco-Martinez, N Rubido, Ch G Antonopoulos, and MS Baptista. 'Successful network inference from time-series data using mutual information rate.' In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 26.4 (2016), p. 043102.

[38] Gérard Biau and Erwan Scornet. 'A random forest guided tour.' In: *Test* 25.2 (2016), pp. 197–227.

[39] Jacob Bien and Robert J Tibshirani. 'Sparse estimation of a covariance matrix.' In: *Biometrika* 98.4 (2011), pp. 807–820.

[40] David M Blei, Andrew Y Ng, and Michael I Jordan. 'Latent dirichlet allocation.' In: *the Journal of machine Learning research* 3 (2003), pp. 993–1022.

[41] John L. G. Board, Charles M. S. Sutcliffe, and William T. Ziemba. 'Portfolio Selection: Markowitz Mean-Variance Model.' In: *Encyclopedia of Optimization*. Ed. by Christodoulos A. Floudas and Panos M. Pardalos. Boston, MA: Springer US, 2001, pp. 1992–1998. ISBN: 978-0-306-48332-5. DOI: 10.1007/0-306-48332-7_391. URL: https://doi.org/10.1007/0-306-48332-7_391.

[42] Paolo Boldi, Marco Rosa, Massimo Santini, and Sebastiano Vigna. 'Layered Label Propagation: A MultiResolution Coordinate-Free Ordering for Compressing Social Networks.' In: *Proceedings of the 20th international conference on World Wide Web*. Ed. by Sadagopan Srinivasan, Krithi Ramamritham, Arun Kumar, M. P. Ravindra, Elisa Bertino, and Ravi Kumar. ACM Press, 2011, pp. 587–596.

[43] Paolo Boldi and Sebastiano Vigna. 'The WebGraph Framework I: Compression Techniques.' In: *Proc. of the Thirteenth International World Wide Web Conference (WWW 2004)*. Manhattan, USA: ACM Press, 2004, pp. 595–601.

[44] Hanen Borchani, Gherardo Varando, Concha Bielza, and Pedro Larrañaga. 'A survey on multi-output regression.' In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 5.5 (2015), pp. 216–233.

[45] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. 'Distributed optimization and statistical learning via the alternating direction method of multipliers.' In: *Foundations and Trends® in Machine learning* 3.1 (2011), pp. 1–122.

[46] Leo Breiman. 'Random forests.' In: *Machine learning* 45.1 (2001), pp. 5–32.

[47] Kristijan Breznik. 'On the gender effects of handedness in professional tennis.' In: *Journal of sports science & medicine* 12.2 (2013), p. 346.

[48] Sergey Brin and Lawrence Page. 'The anatomy of a large-scale hypertextual web search engine.' In: (1998).

[49] Laura F Bringmann, Ellen L Hamaker, Daniel E Vigo, André Aubert, Denny Borsboom, and Francis Tuerlinckx. 'Changing dynamics: Time-varying autoregressive models using generalized additive modeling.' In: *Psychological methods* 22.3 (2017), p. 409.

[50] Andrei Broder, Ravi Kumar, Farzin Maghoul, Prabhakar Raghavan, Sridhar Rajagopalan, Raymie Stata, Andrew Tomkins, and Janet Wiener. 'Graph structure in the web.' In: *Computer networks* 33.1-6 (2000), pp. 309–320.

[51] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 'Grarep: Learning graph representations with global structural information.' In: *Proceedings of the 24th ACM international on conference on information and knowledge management*. 2015, pp. 891–900.

[52] Bruce Carse, Barry Meadows, Roy Bowers, and Philip Rowe. 'Affordable clinical gait analysis: An assessment of the marker tracking accuracy of a new low-cost optical 3D motion analysis system.' In: *Physiotherapy* 99.4 (2013), pp. 347–351.

[53] Claire Chambers, Nidhi Seethapathi, Rachit Saluja, Helen Loeb, Samuel R Pierce, Daniel K Bogen, Laura Prosser, Michelle J Johnson, and Konrad P Kording. 'Computer vision to automatically assess infant neuromotor risk.' In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 28.11 (2020), pp. 2431–2442.

[54] Venkat Chandrasekaran, Pablo A Parrilo, and Alan S Willsky. 'Latent variable graphical model selection via convex optimization.' In: *2010 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE. 2010, pp. 1610–1613.

[55] Venkat Chandrasekaran, Sujay Sanghavi, Pablo A Parrilo, and Alan S Willsky. 'Rank-sparsity incoherence for matrix decomposition.' In: *SIAM Journal on Optimization* 21.2 (2011), pp. 572–596.

[56] Andersen Chang, Tianyi Yao, and Genevera I Allen. 'Graphical Models and Dynamic Latent Factors for Modeling Functional Brain Connectivity.' In: *2019 IEEE Data Science Workshop (DSW)*. IEEE. 2019, pp. 57–63.

[57] Kai Chen, Yi Zhou, and Fangyan Dai. 'A LSTM-based method for stock returns prediction: A case study of China stock market.' In: *2015 IEEE international conference on big data (big data)*. IEEE. 2015, pp. 2823–2824.

[58] Tianqi Chen and Carlos Guestrin. 'Xgboost: A scalable tree boosting system.' In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 2016, pp. 785–794.

[59] Xiaowei Chen, Yongkun Li, Pinghui Wang, and John Lui. 'A general framework for estimating graphlet statistics via random walk.' In: *arXiv preprint arXiv:1603.07504* (2016).

[60] Zexun Chen, Bo Wang, and Alexander N Gorban. 'Multivariate Gaussian and Student-t process regression for multi-output prediction.' In: *Neural Computing and Applications* 32.8 (2020), pp. 3005–3028.

[61] Lulu Cheng, Liang Shan, and Inyoung Kim. 'Multilevel Gaussian graphical model for multilevel networks.' In: *Journal of Statistical Planning and Inference* 190 (2017), pp. 1–14.

[62] Tiberiu Chis and Peter G Harrison. 'Adapting hidden Markov models for online learning.' In: *Electronic Notes in Theoretical Computer Science* 318 (2015), pp. 109–127.

[63] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 'Learning phrase representations using RNN encoder-decoder for statistical machine translation.' In: *arXiv preprint arXiv:1406.1078* (2014).

[64] Myung Jin Choi, Venkat Chandrasekaran, and Alan S Willsky. 'Gaussian multiresolution models: Exploiting sparse Markov and covariance structure.' In: *IEEE Transactions on Signal Processing* 58.3 (2009), pp. 1012–1024.

[65] Myung Jin Choi, Vincent YF Tan, Animashree Anandkumar, and Alan S Willsky. 'Learning latent tree graphical models.' In: *Journal of Machine Learning Research* 12.May (2011), pp. 1771–1812.

[66] Federico Ciech and Veronica Tozzo. *Time Adaptive Gaussian Model*. 2021. arXiv: 2102.01238 [stat.ML].

[67] Aaron Clauset, Cosma Rohilla Shalizi, and Mark EJ Newman. 'Power-law distributions in empirical data.' In: *SIAM review* 51.4 (2009), pp. 661–703.

[68] Aaron Clauset, Maxwell Young, and Kristian Skrede Gleditsch. 'On the frequency of severe terrorist events.' In: *Journal of Conflict Resolution* 51.1 (2007), pp. 58–87.

[69] Peter Clifford. 'Markov random fields in statistics.' In: *Disorder in physical systems: A volume in honour of John M. Hammersley* (1990), pp. 19–32.

[70] Steffi L Colyer, Murray Evans, Darren P Cosker, and Aki IT Salo. 'A review of the evolution of vision-based motion analysis and the integration of advanced computer vision methods towards developing a markerless system.' In: *Sports medicine-open* 4.1 (2018), pp. 1–15.

[71] L da F Costa, Francisco A Rodrigues, Gonzalo Travieso, and Paulino Ribeiro Villas Boas. 'Characterization of complex networks: A survey of measurements.' In: *Advances in physics* 56.1 (2007), pp. 167–242.

[72] Rainer Dahlhaus and Michael Eichler. 'Causality and graphical models in time series analysis.' In: *Oxford Statistical Science Series* (2003), pp. 115–137.

[73] Eswar Damaraju, Elena A Allen, Aysenil Belger, Judith M Ford, S McEwen, DH Mathalon, BA Mueller, GD Pearlson, SG Potkin, A Preda, et al. 'Dynamic functional connectivity analysis reveals transient states of dysconnectivity in schizophrenia.' In: *NeuroImage: Clinical* 5 (2014), pp. 298–308.

[74] Patrick Danaher, Pei Wang, and Daniela M Witten. 'The joint graphical lasso for inverse covariance estimation across multiple classes.' In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 76.2 (2014), pp. 373–397.

[75] HE Daniels. 'The asymptotic efficiency of a maximum likelihood estimator.' In: *Fourth Berkeley Symposium on Mathematical Statistics and Probability*. Vol. 1. University of California Press Berkeley. 1961, pp. 151–163.

[76] Kousik Das, Sovan Samanta, and Madhumangal Pal. 'Study on centrality measures in social networks: a survey.' In: *Social network analysis and mining* 8.1 (2018), p. 13.

[77] Arthur P Dempster. 'Covariance selection.' In: *Biometrics* (1972), pp. 157–175.

[78] Arthur P Dempster, Nan M Laird, and Donald B Rubin. 'Maximum likelihood from incomplete data via the EM algorithm.' In: *Journal of the Royal Statistical Society: Series B (Methodological)* 39.1 (1977), pp. 1–22.

[79] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 'Imagenet: A large-scale hierarchical image database.' In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[80] Yann Desmarais, Denis Mottet, Pierre Slangen, and Philippe Montesinos. 'A review of 3D human pose estimation algorithms for markerless motion capture.' In: *arXiv preprint arXiv:2010.06449* (2020).

[81] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 'Bert: Pre-training of deep bidirectional transformers for language understanding.' In: *arXiv preprint arXiv:1810.04805* (2018).

[82] Tamar Dimitrova, Kristijan Petrovski, and Ljupcho Kocarev. 'Graphlets in multiplex networks.' In: *Scientific reports* 10.1 (2020), pp. 1–13.

[83] Nicholas Dingle, William Knottenbelt, and Demetris Spanias. 'On the (page) ranking of professional tennis players.' In: *Computer Performance Engineering*. Springer, 2012, pp. 237–247.

[84] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 'metapath2vec: Scalable representation learning for heterogeneous networks.' In: *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. 2017, pp. 135–144.

[85] Hongyan Dui, Xueyu Meng, Hui Xiao, and Jianjun Guo. 'Analysis of the cascading failure for scale-free networks based on a multi-strategy evolutionary game.' In: *Reliability Engineering & System Safety* 199 (2020), p. 106919.

[86] Sacha Epskamp, Denny Borsboom, and Eiko I Fried. 'Estimating psychological networks and their accuracy: A tutorial paper.' In: *Behavior Research Methods* 50.1 (2018), pp. 195–212.

[87] Paul Erdős, Alfréd Rényi, et al. 'On the evolution of random graphs.' In: *Publ. Math. Inst. Hung. Acad. Sci* 5.1 (1960), pp. 17–60.

[88] Paul Erdös et al. 'On random graphs.' In: ().

[89] Shimon Even. *Graph algorithms*. Cambridge University Press, 2011.

[90] Brian S Everitt. 'Finite mixture distributions.' In: *Wiley StatsRef: Statistics Reference Online* (2014).

[91] Jesse Fagan, Katherine S Eddens, Jennifer Dolly, Nathan L Vanderford, Heidi Weiss, and Justin S Levens. 'Assessing research collaboration through co-authorship network analysis.' In: *The journal of research administration* 49.1 (2018), p. 76.

[92] Alireza Farasat, Alexander Nikolaev, Sargur N Srihari, and Rachael Hageman Blair. 'Probabilistic graphical models in modern social network analysis.' In: *Social Network Analysis and Mining* 5.1 (2015), pp. 1–18.

[93] Luciano da Fontoura Costa and Roberto Marcond Cesar Jr. *Shape analysis and classification: theory and practice*. CRC press, 2010.

[94] G David Forney. 'The Viterbi algorithm.' In: *Proceedings of the IEEE* 61.3 (1973), pp. 268–278.

[95] Santo Fortunato. 'Community detection in graphs.' In: *Physics reports* 486.3-5 (2010), pp. 75–174.

[96] Santo Fortunato, Carl T Bergstrom, Katy Börner, James A Evans, Dirk Helbing, Staša Milojević, Alexander M Petersen, Filippo Radicchi, Roberta Sinatra, Brian Uzzi, et al. 'Science of science.' In: *Science* 359.6379 (2018), eaao0185.

[97] Nicholas J Foti, Rahul Nadkarni, AK Lee, and Emily B Fox. 'Sparse plus low-rank graphical models of time series for functional connectivity in MEG.' In: *2nd KDD Workshop on Mining and Learning from Time Series*. 2016.

[98] Brendan J Frey. 'Extending factor graphs so as to unify directed and undirected graphical models.' In: *arXiv preprint arXiv:1212.2486* (2012).

[99] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*. Vol. 1. 10. Springer series in statistics New York, 2001.

[100] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. 'Sparse inverse covariance estimation with the graphical lasso.' In: *Biostatistics* 9.3 (2008), pp. 432–441.

[101] Luca Garello, Matteo Moro, Chiara Tacchino, Francesca Campone, Paola Durand, Isabella Blanchi, Paolo Moretti, Maura Casadio, and Francesca Odone. 'A Study of At-term and Preterm Infants' Motion Based on Markerless Video Analysis.' In: (2021).

[102] Sinong Geng, Zhaobin Kuang, Peggy Peissig, and David Page. 'Temporal Poisson Square Root Graphical Models.' In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. Stockholmsmässan, Stockholm Sweden: PMLR, 2018, pp. 1714–1723. URL: http://proceedings.mlr.press/v80/geng18a.html.

[103] Alex J Gibberd and Sandipan Roy. 'Multiple changepoint estimation in high-dimensional gaussian graphical models.' In: *arXiv preprint arXiv:1712.05786* (2017).

[104] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 'Neural message passing for quantum chemistry.' In: *International conference on machine learning*. PMLR. 2017, pp. 1263–1272.

[105] Yoav Goldberg. 'Neural network methods for natural language processing.' In: *Synthesis lectures on human language technologies* 10.1 (2017), pp. 1–309.

[106] Bruce Golden. 'Shortest-path algorithms: A comparison.' In: *Operations Research* 24.6 (1976), pp. 1164–1168.

[107] Clive WJ Granger. 'Investigating causal relations by econometric models and cross-spectral methods.' In: *Econometrica: journal of the Econometric Society* (1969), pp. 424–438.

[108] Aditya Grover and Jure Leskovec. 'node2vec: Scalable feature learning for networks.' In: *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. 2016, pp. 855–864.

[109] Jian Guo, Elizaveta Levina, George Michailidis, and Ji Zhu. 'Joint estimation of multiple graphical models.' In: *Biometrika* 98.1 (2011), pp. 1–15.

[110] Uri Hadar et al. 'High-order hidden Markov models-estimation and implementation.' In: *2009 IEEE/SP 15th Workshop on Statistical Signal Processing*. IEEE. 2009, pp. 249–252.

[111] David Hallac, Jure Leskovec, and Stephen Boyd. 'Network lasso: Clustering and optimization in large graphs.' In: *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*. ACM. 2015, pp. 387–396.

[112] David Hallac, Youngsuk Park, Stephen Boyd, and Jure Leskovec. 'Network inference via the time-varying graphical lasso.' In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. 2017, pp. 205–213.

[113] David Hallac, Sagar Vare, Stephen Boyd, and Jure Leskovec. 'Toeplitz inverse covariance-based clustering of multivariate time series data.' In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. 2017, pp. 215–223.

[114] Will Hamilton, Zhitao Ying, and Jure Leskovec. 'Inductive representation learning on large graphs.' In: *Advances in neural information processing systems* 30 (2017).

[115] William L Hamilton. 'Graph representation learning.' In: *Synthesis Lectures on Artifical Intelligence and Machine Learning* 14.3 (2020), pp. 1–159.

[116] Guyue Han and Harish Sethu. 'Waddling random walk: Fast and accurate mining of motif statistics in large graphs.' In: *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE. 2016, pp. 181–190.

[117] Michael Hardy. 'Pareto's law.' In: *The Mathematical Intelligencer* 32.3 (2010), pp. 38–43.

[118] Adib Hasan, Po-Chien Chung, and Wayne Hayes. 'Graphettes: Constant-time determination of graphlet and orbit identity including (possibly disconnected) graphlets up to size 8.' In: *PloS one* 12.8 (2017), e0181570.

[119] Mohammad Al Hasan and Mohammed J Zaki. 'A survey of link prediction in social networks.' In: *Social network data analytics*. Springer, 2011, pp. 243–275.

[120] Jonas MB Haslbeck and Lourens J Waldorp. 'mgm: Structure estimation for time-varying mixed graphical models in high-dimensional data.' In: *arXiv preprint arXiv:1510.06871* (2015).

[121] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.

[122] Trevor Hastie, Robert Tibshirani, and Martin Wainwright. *Statistical learning with sparsity: the lasso and generalizations*. Chapman and Hall/CRC, 2015.

[123] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 'Deep residual learning for image recognition.' In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[124] Zonglu He and Koichi Maekawa. 'On spurious Granger causality.' In: *Economics Letters* 73.3 (2001), pp. 307–313.

[125] Michael Hecker, Sandro Lambeck, Susanne Toepfer, Eugene Van Someren, and Reinhard Guthke. 'Gene regulatory network inference: data integration in dynamic models—a review.' In: *Biosystems* 96.1 (2009), pp. 86–103.

[126] Dorte Henriksen. 'The rise in co-authorship in the social sciences (1980–2013).' In: *Scientometrics* 107.2 (2016), pp. 455–476.

[127] Nikolas Hesse, Christoph Bodensteiner, Michael Arens, Ulrich G Hofmann, Raphael Weinberger, and A Sebastian Schroeder. 'Computer vision for medical infant motion analysis: State of the art and rgb-d data set.' In: *Proceedings of the ECCV*. 2018, pp. 0–0.

[128] Akinori Higaki, Teruyoshi Uetani, Shuntaro Ikeda, and Osamu Yamaguchi. 'Co-authorship network analysis in cardiovascular research utilizing machine learning (2009–2019).' In: *International Journal of Medical Informatics* 143 (2020), p. 104274.

[129] Tomaž Hočevar and Janez Demšar. 'A combinatorial approach to graphlet counting.' In: *Bioinformatics* 30.4 (2014), pp. 559–565.

[130] Sepp Hochreiter and Jürgen Schmidhuber. 'Long short-term memory.' In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[131] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 'Heterogeneous graph transformer.' In: *Proceedings of The Web Conference 2020*. 2020, pp. 2704–2710.

[132] Lei Huang, Li Liao, and Cathy H Wu. 'Inference of protein-protein interaction networks from multiple heterogeneous data.' In: *EURASIP Journal on Bioinformatics and Systems Biology* 2016.1 (2016), pp. 1–9.

[133] Mian Huang, Yue Huang, and Kang He. 'Estimation and testing non-homogeneity of Hidden Markov model with application in financial time series.' In: *Statistics and Its Interface* 12 (Jan. 2019), pp. 215–225. DOI: 10.4310/SII.2019.v12.n2.a3.

[134] Ali Jalali, Pradeep Ravikumar, Vishvas Vasuki, and Sujay Sanghavi. 'On learning discrete graphical models using group-sparse regularization.' In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. 2011, pp. 378–387.

[135] Jeannette Janssen, Matt Hurshman, and Nauzer Kalyaniwalla. 'Model selection for social networks using graphlets.' In: *Internet Mathematics* 8.4 (2012), pp. 338–363.

[136] David T Jones, Daniel WA Buchan, Domenico Cozzetto, and Massimiliano Pontil. 'PSICOV: precise structural contact prediction using sparse inverse covariance estimation on large multiple sequence alignments.' In: *Bioinformatics* 28.2 (2011), pp. 184–190.

[137] R Kasprzak. 'Diffusion in networks.' In: *Journal of Telecommunications and Information Technology* (2012), pp. 99–106.

[138] Diederik P Kingma and Jimmy Ba. 'Adam: A method for stochastic optimization.' In: *arXiv preprint arXiv:1412.6980* (2014).

[139] Thomas N Kipf and Max Welling. 'Semi-supervised classification with graph convolutional networks.' In: *arXiv preprint arXiv:1609.02907* (2016).

[140] Franc JGM Klaassen and Jan R Magnus. 'Are points in tennis independent and identically distributed? Evidence from a dynamic binary panel data model.' In: *Journal of the American Statistical Association* 96.454 (2001), pp. 500–509.

[141] Franc JGM Klaassen and Jan R Magnus. 'Forecasting the winner of a tennis match.' In: *European Journal of Operational Research* 148.2 (2003), pp. 257–267.

[142] William J Knottenbelt, Demetris Spanias, and Agnieszka M Madurska. 'A common-opponent stochastic model for predicting the outcome of professional tennis matches.' In: *Computers & Mathematics with Applications* 64.12 (2012), pp. 3820–3827.

[143] Oleksii Kuchaiev, Aleksandar Stevanović, Wayne Hayes, and Nataša Pržulj. 'GraphCrunch 2: software tool for network modeling, alignment and clustering.' In: *BMC bioinformatics* 12.1 (2011), pp. 1–13.

[144] Lukas Kuld and John O'Hagan. 'Rise of multi-authored papers in economics: Demise of the 'lone star'and why?' In: *Scientometrics* 114.3 (2018), pp. 1207–1225.

[145] Ajay Kumar, Shashank Sheshar Singh, Kuldeep Singh, and Bhaskar Biswas. 'Link prediction techniques, applications, and performance: A survey.' In: *Physica A: Statistical Mechanics and its Applications* 553 (2020), p. 124289.

[146] Tom Dupré La Tour, Thomas Moreau, Mainak Jas, and Alexandre Gramfort. 'Multivariate convolutional sparse coding for electromagnetic brain signals.' In: *Advances in Neural Information Processing Systems*. 2018, pp. 3292–3302.

[147] Steffen L Lauritzen. *Graphical models*. Vol. 17. Clarendon Press, 1996.

[148] Chengwei Lei and Jianhua Ruan. 'A novel link prediction algorithm for reconstructing protein–protein interaction networks by topological similarity.' In: *Bioinformatics* 29.3 (2013), pp. 355–364.

[149] Jure Leskovec and Rok Sosič. 'SNAP: A General-Purpose Network Analysis and Graph-Mining Library.' In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 8.1 (2016), p. 1.

[150] Jurica Levatić, Michelangelo Ceci, Dragi Kocev, and Sašo Džeroski. 'Semi-supervised learning for multi-target regression.' In: *International workshop on new frontiers in mining complex patterns*. Springer. 2014, pp. 3–18.

[151] F. Li, L. Zhang, B. Chen, D. Gao, Y. Cheng, X. Zhang, Y. Yang, K. Gao, Z. Huang, and J. Peng. 'A Light Gradient Boosting Machine for Remainning Useful Life Estimation of Aircraft Engines.' In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. 2018, pp. 3562–3567.

[152] Xiang Li, Yao Wu, Martin Ester, Ben Kao, Xin Wang, and Yudian Zheng. 'Semi-supervised clustering in attributed heterogeneous information networks.' In: *Proceedings of the 26th international conference on world wide web*. 2017, pp. 1621–1629.

[153] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. 'Gated graph sequence neural networks.' In: *arXiv preprint arXiv:1511.05493* (2015).

[154] Roderick JA Little and Donald B Rubin. *Statistical analysis with missing data*. Vol. 793. Wiley, 2019.

[155] Guoguang Liu. 'An ecommerce recommendation algorithm based on link prediction.' In: *Alexandria Engineering Journal* 61.1 (2022), pp. 905–910.

[156] Qingqing Long, Yilun Jin, Guojie Song, Yi Li, and Wei Lin. 'Graph structural-topic neural network.' In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2020, pp. 1065–1073.

[157] Ilya Loshchilov and Frank Hutter. 'Sgdr: Stochastic gradient descent with warm restarts.' In: *arXiv preprint arXiv:1608.03983* (2016).

[158] Anani Lotsi and Ernst Wit. 'High dimensional sparse gaussian graphical mixture model.' In: *arXiv preprint arXiv:1308.3381* (2013).

[159] László Lovász et al. 'Random walks on graphs: A survey.' In: *Combinatorics, Paul erdos is eighty* 2.1 (1993), pp. 1–46.

[160] Linyuan Lü and Tao Zhou. 'Link prediction in complex networks: A survey.' In: *Physica A: statistical mechanics and its applications* 390.6 (2011), pp. 1150–1170.

[161] Scott M Lundberg and Su-In Lee. 'A unified approach to interpreting model predictions.' In: *Advances in neural information processing systems*. 2017, pp. 4765–4774.

[162] Shang-Min Ma, Chao-Chin Liu, Yue Tan, and Shang-Chun Ma. 'Winning matches in Grand Slam men's singles: An analysis of player performance-related variables from 1991 to 2008.' In: *Journal of sports sciences* 31.11 (2013), pp. 1147–1155.

[163] James MacQueen et al. 'Some methods for classification and analysis of multivariate observations.' In: *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. Vol. 1. 14. Oakland, CA, USA. 1967, pp. 281–297.

[164] William G Madow, Harold Nisselson, and Ingram Olkin. 'Incomplete data in sample surveys. Vol. 1: Report and case studies.' In: (1983).

[165] Víctor Martínez, Fernando Berzal, and Juan-Carlos Cubero. 'A survey of link prediction in complex networks.' In: *ACM computing surveys (CSUR)* 49.4 (2016), pp. 1–33.

[166] Frank J Massey Jr. 'The Kolmogorov-Smirnov test for goodness of fit.' In: *Journal of the American statistical Association* 46.253 (1951), pp. 68–78.

[167] Alexander Mathis, Pranav Mamidanna, Kevin M Cury, Taiga Abe, Venkatesh N Murthy, Mackenzie Weygandt Mathis, and Matthias Bethge. 'DeepLabCut: markerless pose estimation of user-defined body parts with deep learning.' In: *Nature neuroscience* 21.9 (2018), p. 1281.

[168] Brian W Matthews. 'Comparison of the predicted and observed secondary structure of T4 phage lysozyme.' In: *Biochimica et Biophysica Acta (BBA)-Protein Structure* 405.2 (1975), pp. 442–451.

[169] Alberto Maydeu-Olivares and Carlos Garcia-Forero. 'Goodness-of-fit testing.' In: *International encyclopedia of education* 7.1 (2010), pp. 190–196.

[170] L Meinecke, N Breitbach-Faller, C Bartz, R Damen, G Rau, and C Disselhorst-Klug. 'Movement analysis in the early detection of newborns at risk for developing spasticity due to infantile cerebral palsy.' In: *Human movement science* 25.2 (2006), pp. 125–144.

[171] Nicolai Meinshausen, Peter Bühlmann, et al. 'High-dimensional graphs and variable selection with the lasso.' In: *The annals of statistics* 34.3 (2006), pp. 1436–1462.

[172] Zhaoshi Meng, Brian Eriksson, and Al Hero. 'Learning latent variable Gaussian graphical models.' In: *International Conference on Machine Learning*. 2014, pp. 1269–1277.

[173] Umberto Michieli. 'Complex Network Analysis of Men Single ATP Tennis Matches.' In: *arXiv preprint arXiv:1804.08138* (2018).

[174] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 'Efficient estimation of word representations in vector space.' In: *arXiv preprint arXiv:1301.3781* (2013).

[175] Stanley Milgram. 'The small world problem.' In: *Psychology today* 2.1 (1967), pp. 60–67.

[176] Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. 'Network motifs: simple building blocks of complex networks.' In: *Science* 298.5594 (2002), pp. 824–827.

[177] David Mimno, Hanna Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. 'Optimizing semantic coherence in topic models.' In: *Proceedings of the 2011 conference on empirical methods in natural language processing*. 2011, pp. 262–272.

[178] Karthik Mohan, Mike Chung, Seungyeop Han, Daniela Witten, Su-In Lee, and Maryam Fazel. 'Structured learning of Gaussian graphical models.' In: *Advances in neural information processing systems*. 2012, pp. 620–628.

[179] Evan J Molinelli, Anil Korkut, Weiqing Wang, Martin L Miller, Nicholas P Gauthier, Xiaohong Jing, Poorvi Kaushik, Qin He, Gordon Mills, David B Solit, et al. 'Perturbation biology: inferring signaling networks in cellular systems.' In: *PLoS computational biology* 9.12 (2013), e1003290.

[180] Ricardo Pio Monti, Peter Hellyer, David Sharp, Robert Leech, Christoforos Anagnostopoulos, and Giovanni Montana. 'Estimating time-varying brain connectivity networks from functional MRI time series.' In: *NeuroImage* 103 (2014), pp. 427–443.

[181] Stefano Mossa, Marc Barthélémy, H. Eugene Stanley, and Luís A. Nunes Amaral. 'Truncation of Power Law Behavior in "Scale-Free" Network Models due to Information Filtering.' In: *Phys. Rev. Lett.* 88 (13 2002), p. 138701. DOI: 10.1103/PhysRevLett.88.138701. URL: https://link.aps.org/doi/10.1103/PhysRevLett.88.138701.

[182] Sadegh Motallebi, Sadegh Aliakbary, and Jafar Habibi. 'Generative model selection using a scalable and size-independent complex network classifier.' In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 23.4 (2013), p. 043127.

[183] Ryan L Murphy, Balasubramaniam Srinivasan, Vinayak Rao, and Bruno Ribeiro. 'Janossy pooling: Learning deep permutation-invariant functions for variable-size inputs.' In: *arXiv preprint arXiv:1811.01900* (2018).

[184] Alberto Natali, Mario Coutino, Elvin Isufi, and Geert Leus. *Online Time-Varying Topology Identification via Prediction-Correction Algorithms*. 2020. arXiv: 2010.11634 [eess.SP].

[185] John Ashworth Nelder and Robert WM Wedderburn. 'Generalized linear models.' In: *Journal of the Royal Statistical Society: Series A (General)* 135.3 (1972), pp. 370–384.

[186] Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. 'ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing.' In: *Proceedings of the 18th BioNLP Workshop and Shared Task*. Florence, Italy: Association for Computational Linguistics, Aug. 2019, pp. 319–327. DOI: 10.18653/v1/W19-5034. eprint: arXiv:1902.07669. URL: https://www.aclweb.org/anthology/W19-5034.

[187] Mark EJ Newman. 'Models of the small world.' In: *Journal of Statistical Physics* 101.3 (2000), pp. 819–841.

[188] Mark EJ Newman. 'Power laws, Pareto distributions and Zipf's law.' In: *Contemporary physics* 46.5 (2005), pp. 323–351.

[189] Andrew Y Ng, Michael I Jordan, and Yair Weiss. 'On spectral clustering: Analysis and an algorithm.' In: *Advances in neural information processing systems*. 2002, pp. 849–856.

[190] Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. 'Holographic embeddings of knowledge graphs.' In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 30. 1. 2016.

[191] Maria Rosa Nieto and Esther Ruiz. 'Frontiers in VAR forecasting and backtesting.' In: *International Journal of Forecasting* 32.2 (2016), pp. 475–501.

[192] A James O'Malley. 'Probability formulas and statistical analysis in tennis.' In: *Journal of Quantitative Analysis in Sports* 4.2 (2008).

[193] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 'Asymmetric transitivity preserving graph embedding.' In: *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. 2016, pp. 1105–1114.

[194] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. *The PageRank citation ranking: Bringing order to the web.* Tech. rep. Stanford InfoLab, 1999.

[195] Shashank Pandit, Duen Horng Chau, Samuel Wang, and Christos Faloutsos. 'Netprobe: a fast and scalable system for fraud detection in online auction networks.' In: *Proceedings of the 16th international conference on World Wide Web*. 2007, pp. 201–210.

[196] Angeliki Papana, Catherine Kyrtsou, Dimitris Kugiumtzis, and Cees Diks. 'Detecting causality in non-stationary time series using partial symbolic transfer entropy: Evidence in financial data.' In: *Computational economics* 47.3 (2016), pp. 341–365.

[197] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 'Pytorch: An imperative style, high-performance deep learning library.' In: *Advances in neural information processing systems* 32 (2019).

[198] Martin Pelikan, David E Goldberg, Erick Cantú-Paz, et al. 'BOA: The Bayesian optimization algorithm.' In: *Proceedings of the genetic and evolutionary computation conference GECCO-99*. Vol. 1. Citeseer. 1999, pp. 525–532.

[199] David M Pennock, Gary W Flake, Steve Lawrence, Eric J Glover, and C Lee Giles. 'Winners don't take all: Characterizing the competition for links on the web.' In: *Proceedings of the national academy of sciences* 99.8 (2002), pp. 5207–5211.

[200] José Pereira, Morteza Ibrahimi, and Andrea Montanari. 'Learning networks of stochastic differential equations.' In: *Advances in Neural Information Processing Systems*. 2010, pp. 172–180.

[201] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 'Deepwalk: Online learning of social representations.' In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2014, pp. 701–710.

[202] Joss Peters. 'Predicting the Outcomes of Professional Tennis Matches.' In: ().

[203] Steven Peterson. *Investment Theory and Risk Management,+ Website*. Vol. 711. John Wiley & Sons, 2012.

[204] Niklas Pfister, Stefan Bauer, and Jonas Peters. 'Learning stable and predictive structures in kinetic systems.' In: *Proceedings of the National Academy of Sciences* 116.51 (2019), pp. 25405–25411.

[205] Trang Pham, Truyen Tran, Dinh Phung, and Svetha Venkatesh. 'Column networks for collective classification.' In: *Thirty-first AAAI conference on artificial intelligence*. 2017.

[206] Nataša Pržulj. 'Biological network comparison using graphlet degree distribution.' In: *Bioinformatics* 23.2 (2007), e177–e183.

[207] Natasa Pržulj, Derek G Corneil, and Igor Jurisica. 'Modeling interactome: scale-free or geometric?' In: *Bioinformatics* 20.18 (2004), pp. 3508–3515.

[208] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. 'Pointnet: Deep learning on point sets for 3d classification and segmentation.' In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 652–660.

[209] Filippo Radicchi. 'Who is the best player ever? A complex network analysis of the history of professional tennis.' In: *PloS one* 6.2 (2011), e17249.

[210] Garvesh Raskutti, Bin Yu, Martin J Wainwright, and Pradeep K Ravikumar. 'Model Selection in Gaussian Graphical Models: High-Dimensional Consistency of ℓ_1-regularized MLE.' In: *Advances in Neural Information Processing Systems*. 2009, pp. 1329–1336.

[211] Pradeep Ravikumar, Martin J Wainwright, John D Lafferty, et al. 'High-dimensional Ising model selection using ℓ_1-regularized logistic regression.' In: *The Annals of Statistics* 38.3 (2010), pp. 1287–1319.

[212] Pradeep Ravikumar, Martin J Wainwright, Garvesh Raskutti, Bin Yu, et al. 'High-dimensional covariance estimation by minimizing ℓ_1-penalized log-determinant divergence.' In: *Electronic Journal of Statistics* 5 (2011), pp. 935–980.

[213] Sara Rebagliati and Emanuela Sasso. 'Pattern recognition using hidden Markov models in financial time series.' In: *Acta et Commentationes Universitatis Tartuensis de Mathematica* 21.1 (2017), pp. 25–41.

[214] Sidney I Resnick. *Heavy-tail phenomena: probabilistic and statistical modeling*. Springer Science & Business Media, 2007.

[215] Daniel Revuz. *Markov chains*. Elsevier, 2008.

[216] Michael Röder, Andreas Both, and Alexander Hinneburg. 'Exploring the space of topic coherence measures.' In: *Proceedings of the eighth ACM international conference on Web search and data mining*. 2015, pp. 399–408.

[217] Andrew Rosenberg and Julia Hirschberg. 'V-measure: A conditional entropy-based external cluster evaluation measure.' In: *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*. 2007, pp. 410–420.

[218] Ryan A. Rossi and Nesreen K. Ahmed. 'The Network Data Repository with Interactive Graph Analytics and Visualization.' In: *AAAI*. 2015. URL: https://networkrepository.com.

[219] Gerard Salton and Donna Harman. 'Information retrieval.' In: *Encyclopedia of computer science*. 2003, pp. 858–863.

[220] Ricardo Barros Sampaio, Marcus Vinicius de Araújo Fonseca, Fabio Zicker, et al. 'Co-authorship network analysis in health research: method and potential use.' In: *Health research policy and systems* 14.1 (2016), pp. 1–10.

[221] Anida Sarajlić, Noël Malod-Dognin, Ömer Nebil Yaveroğlu, and Nataša Pržulj. 'Graphlet-based characterization of directed networks.' In: *Scientific reports* 6 (2016), p. 35098.

[222] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 'The graph neural network model.' In: *IEEE transactions on neural networks* 20.1 (2008), pp. 61–80.

[223] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 'Modeling relational data with graph convolutional networks.' In: *European semantic web conference*. Springer. 2018, pp. 593–607.

[224] Gideon Schwarz. 'Estimating the Dimension of a Model.' In: *Ann. Statist.* 6.2 (Mar. 1978), pp. 461–464. DOI: 10.1214/aos/1176344136. URL: https://doi.org/10.1214/aos/1176344136.

[225] Olivier Serrat. 'Social network analysis.' In: *Knowledge solutions*. Springer, 2017, pp. 39–43.

[226] Jingbo Shang, Meng Qu, Jialu Liu, Lance M Kaplan, Jiawei Han, and Jian Peng. 'Meta-path guided embedding for similarity search in large-scale heterogeneous information networks.' In: *arXiv preprint arXiv:1610.09769* (2016).

[227] Eric de Silva and Michael PH Stumpf. 'Complex networks and simple models in biology.' In: *Journal of the Royal Society Interface* 2.5 (2005), pp. 419–430.

[228] Christopher A Sims. 'Macroeconomics and reality.' In: *Econometrica: journal of the Econometric Society* (1980), pp. 1–48.

[229] Michal Sipko and William Knottenbelt. 'Machine learning for the prediction of professional tennis matches.' In: *MEng computing-final year project, Imperial College London* (2015).

[230] Stephen M Smith, Karla L Miller, Gholamreza Salimi-Khorshidi, Matthew Webster, Christian F Beckmann, Thomas E Nichols, Joseph D Ramsey, and Mark W Woolrich. 'Network modelling methods for FMRI.' In: *Neuroimage* 54.2 (2011), pp. 875–891.

[231] Linda Sommerlade, Marco Thiel, Bettina Platt, Andrea Plano, Gernot Riedel, Celso Grebogi, Jens Timmer, and Björn Schelter. 'Inference of Granger causal time-dependent influences in noisy multivariate time series.' In: *Journal of neuroscience methods* 203.1 (2012), pp. 173–185.

[232] Eleftherios Spyromitros-Xioufis, Grigorios Tsoumakas, William Groves, and Ioannis Vlahavas. 'Multi-label classification methods for multi-target regression.' In: *arXiv:1211.6581* (2012).

[233] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 'Dropout: a simple way to prevent neural networks from overfitting.' In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958.

[234] Nicolas Städler and Peter Bühlmann. 'Missing values: sparse inverse covariance estimation and an extension to sparse regression.' In: *Statistics and Computing* 22.1 (2012), pp. 219–235.

[235] Oliver Stegle, Sarah A Teichmann, and John C Marioni. 'Computational and analytical challenges in single-cell transcriptomics.' In: *Nature Reviews Genetics* 16.3 (2015), p. 133.

[236] Michael PH Stumpf and Mason A Porter. 'Critical truths about power laws.' In: *Science* 335.6069 (2012), pp. 665–666.

[237] Winfried Stute, Wenceslao Gonzáles Manteiga, and Manuel Presedo Quindimil. 'Bootstrap based goodness-of-fit-tests.' In: *Metrika* 40.1 (1993), pp. 243–256.

[238] Yizhou Sun and Jiawei Han. 'Mining heterogeneous information networks: principles and methodologies.' In: *Synthesis Lectures on Data Mining and Knowledge Discovery* 3.2 (2012), pp. 1–159.

[239] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. 'Pathsim: Meta path-based top-k similarity search in heterogeneous information networks.' In: *Proceedings of the VLDB Endowment* 4.11 (2011), pp. 992–1003.

[240] Robert Tibshirani. 'Regression shrinkage and selection via the lasso.' In: *Journal of the Royal Statistical Society: Series B (Methodological)* 58.1 (1996), pp. 267–288.

[241] Federico Tomasi, Veronica Tozzo, and Annalisa Barla. 'Temporal Pattern Detection in Time-Varying Graphical Models.' In: *Under review* (2020).

[242] Federico Tomasi, Veronica Tozzo, Saverio Salzo, and Alessandro Verri. 'Latent variable time-varying network inference.' In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM. 2018, pp. 2338–2346.

[243] Veronica Tozzo and Annalisa Barla. 'Multi-parameters Model Selection for Network Inference.' In: *International Conference on Complex Networks and Their Applications*. Springer. 2019, pp. 566–577.

[244] Kun Tu, Jian Li, Don Towsley, Dave Braines, and Liam D Turner. 'gl2vec: Learning feature representation using graphlets for directed networks.' In: *Proceedings of the 2019 IEEE/ACM international conference on advances in social networks analysis and mining*. 2019, pp. 216–221.

[245] Marta Tuninetti, Alberto Aleta, Daniela Paolotti, Yamir Moreno, and Michele Starnini. 'Prediction of new scientific collaborations through multiplex networks.' In: *EPJ Data Science* 10.1 (2021), p. 25.

[246] Vladimir Vapnik and Akshay Vashist. 'A new learning paradigm: Learning using privileged information.' In: *Neural networks* 22.5-6 (2009), pp. 544–557.

[247] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 'Attention is all you need.' In: *Advances in neural information processing systems* 30 (2017).

[248] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 'Graph attention networks.' In: *stat* 1050 (2017), p. 20.

[249] Jean-Philippe Vert, Koji Tsuda, and Bernhard Schölkopf. 'A primer on kernel methods.' In: *Kernel methods in computational biology* 47 (2004), pp. 35–70.

[250] M Vijaikumar, Shirish Shevade, and M Narasimha Murty. 'SoRecGAT: Leveraging graph attention mechanism for top-N social recommendation.' In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2019, pp. 430–446.

[251] Martin J Wainwright, Michael I Jordan, et al. 'Graphical models, exponential families, and variational inference.' In: *Foundations and Trends® in Machine Learning* 1.1–2 (2008), pp. 1–305.

[252] Martin J Wainwright, John D Lafferty, and Pradeep K Ravikumar. 'High-Dimensional Graphical Model Selection Using ℓ_1-Regularized Logistic Regression.' In: *Advances in neural information processing systems*. 2007, pp. 1465–1472.

[253] Jin Wang, Zhiliang Chen, Kaiwei Sun, Hang Li, and Xin Deng. 'Multi-target regression via target specific features.' In: *Knowledge-Based Systems* 170 (2019), pp. 70–78.

[254] Minjie Wang, Lingfan Yu, Da Zheng, Quan Gan, Yu Gai, Zihao Ye, Mufei Li, Jinjing Zhou, Qi Huang, Chao Ma, et al. 'Deep Graph Library: Towards Efficient and Scalable Deep Learning on Graphs.' In: (2019).

[255] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 'Heterogeneous graph attention network.' In: *The world wide web conference*. 2019, pp. 2022–2032.

[256] Duncan J Watts and Steven H Strogatz. 'Collective dynamics of 'small-world'networks.' In: *nature* 393.6684 (1998), pp. 440–442.

[257] Kilian Q Weinberger, Fei Sha, Qihui Zhu, and Lawrence K Saul. 'Graph Laplacian regularization for large-scale semidefinite programming.' In: *Advances in neural information processing systems*. 2007, pp. 1489–1496.

[258] World-Health-Organization. *Preterm birth*. https://www.who.int/news-room/fact-sheets/detail/preterm-birth. 2018.

[259] Shunxin Xiao, Shiping Wang, Yuanfei Dai, and Wenzhong Guo. 'Graph neural networks in node classification: survey and evaluation.' In: *Machine Vision and Applications* 33.1 (2022), pp. 1–19.

[260] Eunho Yang, Genevera Allen, Zhandong Liu, and Pradeep K. Ravikumar. 'Graphical Models via Generalized Linear Models.' In: *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger. Curran Associates, Inc., 2012, pp. 1358–1366. URL: http://papers.nips.cc/paper/4617-graphical-models-via-generalized-linear-models.pdf.

[261] Eunho Yang, Pradeep Ravikumar, Genevera I Allen, and Zhandong Liu. 'Graphical models via univariate exponential family distributions.' In: *The Journal of Machine Learning Research* 16.1 (2015), pp. 3813–3847.

[262] Jilei Yang and Jie Peng. *Estimating Time-Varying Graphical Models*. 2018. arXiv: 1804.03811 [stat.ML].

[263] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 'Xlnet: Generalized autoregressive pretraining for language understanding.' In: *Advances in neural information processing systems* 32 (2019).

[264] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 'Graph convolutional neural networks for web-scale recommender systems.' In: *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 2018, pp. 974–983.

[265] Le Yu, Leilei Sun, Bowen Du, Chuanren Liu, Weifeng Lv, and Hui Xiong. 'Heterogeneous graph representation learning with relation awareness.' In: *IEEE Transactions on Knowledge and Data Engineering* (2022).

[266] Ming Yuan. 'Discussion: Latent variable graphical model selection via convex optimization.' In: *The Annals of Statistics* 40.4 (2012), pp. 1968–1972.

[267] Ming Yuan and Yi Lin. 'Model selection and estimation in the Gaussian graphical model.' In: *Biometrika* 94.1 (2007), pp. 19–35.

[268] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. 'Deep Sets.' In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper/2017/file/f22e4747da1aa27e363d86d40ff442fe-Paper.pdf.

[269] Fanjin Zhang, Xiao Liu, Jie Tang, Yuxiao Dong, Peiran Yao, Jie Zhang, Xiaotao Gu, Yan Wang, Bin Shao, Rui Li, et al. 'Oag: Toward linking large-scale heterogeneous entity graphs.' In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2019, pp. 2585–2595.

[270] Muhan Zhang and Yixin Chen. 'Link prediction based on graph neural networks.' In: *Advances in neural information processing systems* 31 (2018).

[271] Yiming Zhang, Lingfei Wu, Qi Shen, Yitong Pang, Zhihua Wei, Fangli Xu, Ethan Chang, and Bo Long. 'Graph Learning Augmented Heterogeneous Graph Neural Network for Social Recommendation.' In: *arXiv preprint arXiv:2109.11898* (2021).

[272] Yingjian Zhang. 'Prediction of financial time series with Hidden Markov Models.' PhD thesis. Applied Sciences: School of Computing Science, 2004.

[273] Alexandre d'Aspremont. 'Identifying small mean-reverting portfolios.' In: *Quantitative Finance* 11.3 (2011), pp. 351–364.

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

Genova, Italy
July 2022

Davide Garbarino