UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA INDUSTRIAL

COMMUNITY DETECTION IN DYNAMIC ATTRIBUTED NETWORKS

TESIS PARA OPTAR AL GRADO DE
DOCTOR EN SISTEMAS DE INGENIERÍA

RENNY JAVIER MÁRQUEZ CONTRERAS

PROFESOR GUÍA:
RICHARD WEBER HAAS

MIEMBROS DE LA COMISIÓN:
SEBASTIÁN RÍOS PÉREZ
SEBASTIÁN MALDONADO ALARCÓN
ANDRÉ PONCE DE LEÓN FERREIRA DE CARVALHO

Este trabajo ha sido parcialmente financiado por ANID BECAS/Doctorado Nacional/2015
#21151545

SANTIAGO DE CHILE
2024

RESUMEN DE LA TESIS PARA OPTAR
AL GRADO DE DOCTOR EN SISTEMAS DE INGENIERÍA
POR: RENNY JAVIER MÁRQUEZ CONTRERAS
FECHA: 2024
PROF. GUÍA: RICHARD WEBER HAAS

DETECCIÓN DE COMUNIDADES EN REDES DINÁMICAS CON ATRIBUTOS

El análisis de redes sociales puede ser útil para respaldar procesos de toma de decisiones. Una de las herramientas clave para lograr esto es la detección de comunidades. Este enfoque permite identificar grupos, principalmente en redes estáticas donde las conexiones entre los nodos están disponibles. Sin embargo, los problemas del mundo real suelen estar caracterizados por comportamientos que cambian con el tiempo. En estos casos, se necesitan algoritmos de detección de comunidades dinámicas, ya que capturan mejor la dinámica subyacente. Un paso adicional en esta dirección es incluir información sobre los atributos de los nodos y detectar grupos en redes dinámicas que evolucionan, lo que permite obtener resultados más precisos. Además, la capacidad de detectar comunidades *overlapping* representa una mejora, ya que los nodos pueden pertenecer a varios grupos al mismo tiempo. También es relevante que un enfoque pueda detectar automáticamente el número de grupos. Esta tesis presenta dos modelos para la detección de comunidades en redes dinámicas con atributos.

El primer modelo, denominado *COmmunity DEtection in Dynamic Attributed NETworks* (CoDeDANet), consta de dos fases. En la primera fase, basada en *spectral clustering*, se optimiza la relevancia de los atributos en un enfoque que combina los atributos de los nodos con la estructura topológica. En la segunda fase, se utilizan tensores para considerar tanto la información actual como la de instantes de tiempo recientes. Este algoritmo detecta grupos disjuntos, y el número de comunidades es un parámetro definido por el usuario.

El segundo modelo, *Overlapping COmmunity DEtection in Dynamic Attributed NETworks* (OCoDeDANet), utiliza la factorización de matrices no negativas en un enfoque probabilístico para detectar comunidades disjuntas y *overlapping* mediante un algoritmo iterativo que maximiza la probabilidad a posteriori dadas las observaciones. En este caso, el propio algoritmo determina el número de grupos.

Ambos enfoques fueron probados en varias redes sintéticas y datos del mundo real. Los resultados muestran que nuestros modelos superan a distintos algoritmos de la literatura. El uso de la evolución de las redes y los atributos de los nodos en nuestros enfoques condujo a la identificación de comunidades más precisas.

ABSTRACT OF THE THESIS
FOR THE DEGREE OF DOCTOR EN SISTEMAS DE INGENIERÍA
BY: RENNY JAVIER MÁRQUEZ CONTRERAS
DATE: 2024
ADVISOR: RICHARD WEBER HAAS

COMMUNITY DETECTION IN DYNAMIC ATTRIBUTED NETWORKS

The analysis of social networks can be helpful to support policy and decision-making processes. One of the tools to achieve this task is community detection. This approach allows the detection of groups, mostly on static networks where the links between nodes are available. Real-world problems, however, are often characterized by behavior that changes over time. We need dynamic community detection algorithms in such cases because they better capture the underlying dynamics. A step further in this sense is to include attribute information about the nodes and detect groups on dynamic networks that evolve to obtain more accurate results. Also, being able to detect overlapping communities offers an improvement since nodes can belong to different groups at the same time. Furthermore, the capability of an approach to automatically detect the number of groups is also relevant. This thesis presents two models for community detection in dynamic attributed networks.

The first model, for COmmunity DEtection in Dynamic Attributed NETworks (CoDeDANet), comprises two phases. In the first phase, based on spectral clustering, the attributes' importance is optimized in a setting that joins the nodes' features with a topological structure. In the second phase, tensors are used to consider current and past information. This algorithm detects disjoint groups, and the number of communities is a parameter of the approach.

The second model, for Overlapping COmmunity DEtection in Dynamic Attributed NETworks (OCoDeDANet), uses non-negative matrix factorization in a probabilistic approach to detect disjoint and overlapping communities in an iterative algorithm that maximizes the model posterior given the observations. In this case, the algorithm itself determines the number of groups with an automatic relevance determination process.

Both approaches were tested on several synthetic and real-world networks. Results show that our models outperform state-of-the-art algorithms. The use of networks' evolution and nodes' attributes in our approaches led to more accurate communities.

To Mom and Dad, who have always been there for us with unwavering love and support.

Para Mamá y Papá, quienes siempre han estado presentes para nosotros con apoyo incondicional y amor infinito.

Acknowledgment

As I reach the end of this long journey, I want to thank some special people who have been with me through the highs and lows of my PhD path:

To Rafa and Lis, who have been more than friends; you became family many years ago.

To my beloved wife, who has walked this journey alongside me, enduring the challenges, lifting my spirits, and bringing a smile when I needed it most.

To my academic peers and colleagues, who became brothers and sisters: Dana, Edu2, Cristiam, Edu, Vero, Feres, Vale. It was a privilege to share this journey with you, to support each other, to laugh together, and to build memories that will last a lifetime.

To every other PhD student who helped along the way: Andrea, Álvaro, Javier, Coni, Lucho, Ricardo, Sebas. Your support was also part of this achievement.

To the professors I had throughout the PhD at DSI, each of you played a role in my learning and growth, helping to shape my academic path.

To André at Universidade do São Paulo (funded by Universidad de Chile through Ayuda para Estadías Cortas de Investigación) and Mark at McGill University (funded by the Canadian Bureau for International Education through the Emerging Leaders in the Americas Program), thank you for welcoming me and providing internships that were both transformative and enlightening experiences.

To Richard, my advisor, because despite the ups and downs, we were able to get to the finish line.

To Mom, Dad, Richard, Liz, Robert, and Ronald, for all the love and support throughout my academic journey.

And to everyone involved in this achievement whom I may have overlooked in these lines—thank you. This accomplishment is a testament to the collective support and kindness of all those who believed in me.

Table of Content

List of Tables

List of Figures

Introduction

Clustering has become an essential technique in data analysis and machine learning, widely applied for exploratory data analysis across disciplines, including statistics, computer science, biology, and social sciences or psychology (Von Luxburg, 2007). It groups similar objects into clusters based on defined similarity measures, uncovering underlying patterns in complex datasets. Clustering plays a significant role in network analysis, where relationships between objects (nodes) are captured through connections (links). Networks provide a robust framework for representing and understanding complex phenomena across diverse domains (Newman, 2018).

These networks, often composed of social, biological, or information systems, enable us to capture interactions among entities in a simplified yet insightful manner. For example, social networks reveal patterns in friendships, professional connections and other types of relationships (e.g., Facebook, Twitter, contact tracing networks) (Ferrara, 2012; Ferrara et al., 2014; Ozer et al., 2016), information networks map citations and collaborative efforts (e.g., the World Wide Web, citation networks) (Girvan & Newman, 2002; Chakraborty & Chakraborty, 2013; Moradi-Jamei et al., 2021), and biological networks trace relationships within ecosystems (e.g., food webs, protein–protein interactions) (Palla et al., 2005; Wu et al., 2014; Mahmoud et al., 2014).

Many networks evolve dynamically as relationships change over time, such as the formation or dissolution of social ties (social networks) and shifts in citation relevance (citation networks). While valuable, static representations often oversimplify these systems, limiting insights into their temporal evolution. Analyzing networks dynamically allows us to more accurately capture these transitions, providing a realistic representation of interactions and improving our understanding of underlying processes.

In network analysis, clustering is applied as community detection, identifying communities within networks. Here, clusters represent groups of nodes that are more densely connected to each other than to the rest of the network. Communities provide insights into complex systems' structure and behavior, which are critical for understanding aspects such as social behavior, biological functions, technological systems, and information flows.

Traditionally, community detection has focused on static networks. However, because real-world networks evolve, there is a growing need for methods that capture communities' birth, growth, merging, or dissolution over time.

In addition to evolving relationships, network nodes often have attributes, such as demo-

graphic or behavioral information, which add complexity and relevance in identifying meaningful communities. Attributes are particularly useful, as they can reflect homophily (the tendency for similar individuals to connect) and social influence (the impact individuals have on each other). In social networks, for example, these attributes might represent demographic information such as age, gender, or profession, as well as behavioral data.

When these node attributes are incorporated alongside network structure, community detection becomes more effective (Bothorel et al., 2015; Chunaev, 2020). This aspect is enhanced in dynamic networks where both topology and attributes may vary over time. For example, in a network of a bank's company clients, links may be defined by transactions among companies, but features such as the area of the company, region of operations, and time since creation are also important for the bank to classify its clients.

The communities are usually disjoint because each node belongs to only one of the groups found. Nevertheless, nodes in different kinds of networks tend to connect with more than one group. For example, a person can have a relationship with his family, friends, and colleagues; a researcher can be interested in topics from different fields (Xie et al., 2013; Fortunato & Hric, 2016). Therefore, a more thorough network analysis can include overlapping communities, where each node can be a member of more than one group.

When nodes belong to multiple groups, the community assignment is referred to as a *cover*, instead of a traditional *partition* where each node belongs to a single community. Overlapping communities can be *crisp* (no membership degree) or *fuzzy* (varying degrees of membership), capturing the more complex structures present in real-world networks (Fortunato & Hric, 2016).

As relevant as considering these factors is the methodology used to integrate them into designing a better and more helpful community detection algorithm.

Methods for community detection

The methods used for designing the community detection approaches in this thesis are described next.

Spectral clustering

Spectral clustering is a graph-based clustering technique that uses eigenvalues of a similarity matrix to reduce dimensions before applying clustering methods. The main tools for spectral clustering are graph Laplacian matrices, which representation can vary according to the problem being solved (Von Luxburg, 2007).

This technique is especially effective in networks, as it captures community structure by analyzing node connections. First, the similarity or adjacency matrix of the graph is computed. Next, the Laplacian matrix is constructed and decomposed to obtain the eigenvalues and eigenvectors. The key eigenvectors are then used to map nodes into a lower-dimensional space, where traditional clustering methods, such as k-means, can be applied to group similar nodes (Von Luxburg, 2007).

Spectral clustering's use of graph Laplacian matrices allows it to capture complex community structures, making it particularly suited for identifying meaningful communities in dynamic networks.

Tensor representations

A tensor is a multidimensional array. An Nth-order tensor is an element of the tensor product of N vector spaces, each of which has its coordinate system (Kolda & Bader, 2009). This format makes it suitable to represent networks as a Nth-order tensor, where two dimensions correspond to a similarity, adjacency, or Laplacian matrix at a time step, and the third dimension is the number of snapshots that are being used in the representation. In this way, tensors can encode network evolution, capturing changes in structure and attributes, which makes it a valuable tool for community detection in dynamic networks.

To solve the problem, this can be written as a Tucker decomposition, which is a form of higher-order principal component analysis (Kolda & Bader, 2009). Through an iterative algorithm, the tensor decomposition can find matrices from which clustering methods, such as k-means, can be applied to infer communities.

Tensor representations, through methods like Tucker decomposition, provide an efficient way to capture evolving network structure and attributes, identifying both stable and emerging communities.

Probabilistic non-negative matrix factorization

Non-negative matrix factorization (NMF), formally proposed by Lee & Seung (1999), is a classical low-rank matrix factorization model. It is especially applicable for analyzing matrices whose elements are all non-negative (He et al., 2022).

The main idea is to find two non-negative matrices ($\mathbf{W}$ and $\mathbf{H}$) that, when multiplied, reconstruct the original matrix ($\mathbf{V}$). An objective function, such as the square of the Frobenius norm or Kullback–Leibler divergence, must be defined to quantify an approximation error whose goal is to be minimized (He et al., 2022).

The generative capability of NMF can give a good interpretation of community structure when applied to networks, where $\mathbf{V}$ is an adjacency matrix or another representation of the graph. An element of the reconstructed matrix, $v_{\mathrm{i}j}$, can be treated as the expected interaction between nodes i and j. Suppose $v_{\mathrm{i}j}$ is Poisson distributed. In that case, the total expected interactions can be defined as $\hat{v}_{\mathrm{i}j,t} = \sum_{k=1}^{K} w_{\mathrm{i}k,t} h_{kj,t}$, which are obtained by the mutual participation of nodes in community k. As nodes i and j share more communities, they have more interactions, which will result in a higher probability that they will be connected; therefore, nodes in the same community are densely connected (He et al., 2022).

In dynamic networks, the generative nature of probabilistic NMF provides a flexible framework for community membership, enabling nodes to belong to multiple communities as the network evolves over time.

Based on these community detection methods, we now turn to this thesis's specific cha-

llenges and contributions.

Thesis overview and contributions

Given the dynamic nature of many real-world networks and the importance of node attributes, this thesis presents two distinct models for community detection in dynamic attributed networks, addressing limitations in existing approaches, which usually analyze evolving network structure and changing node attributes separately. Both models are designed to detect communities by considering the network's temporal evolution alongside the nodes' attributes, providing a more accurate representation of real-world systems.

Disjoint community detection in dynamic attributed networks

The first model, COmmunity DEtection in Dynamic Attributed NETworks (CoDeDANet), introduces a novel approach for detecting disjoint communities in networks where both the structure and node attributes evolve over time. This model combines spectral clustering to optimize the importance of node attributes with tensor-based methods that incorporate current and past network information. The number of communities is a parameter for this algorithm. The model is tested on both synthetic and real-world networks, demonstrating its ability to outperform state-of-the-art community detection algorithms. The results highlight the benefits of considering both the temporal and attribute-based dynamics of the network, leading to more accurate and insightful community detection.

Overlapping community detection in dynamic attributed networks

The second model, Overlapping COmmunity DEtection in Dynamic Attributed NETworks (OCoDeDANet), extends the capabilities of CoDeDANet by allowing for the detection of both disjoint and overlapping communities. This algorithm uses a probabilistic non-negative matrix factorization approach, solved with an iterative solution algorithm that maximizes the model posterior to detect disjoint and overlapping communities in dynamic attributed networks. This model also incorporates automatic relevance determination to detect the number of communities. The experimental results, tested on synthetic attributed networks and a real-world attributed network, show that OCoDeDANet performs better than existing methods, particularly in cases where the community structure becomes more ambiguous based solely on the network topology. The inclusion of node attributes in these cases helps clarify the community structure.

Submitted works

As a result of this thesis, the following works have been submitted:

1. Márquez, R., Weber, R., and De Carvalho, A. C. (2019). A non-negative matrix factorization approach to update communities in temporal networks using node features. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pp. 728-732. `https://doi.org/10.1145/3341161.3343677` (*Published*).

2. Márquez, R. (2020). Overlapping community detection in static and dynamic networks. In *Proceedings of the 13th International Conference on Web Search and Data Mining (WSDM)*, pp. 925-926. https://doi.org/10.1145/3336191.3372185 (*Published*).
3. Márquez, R. and Weber, R. (2023). Dynamic community detection including node attributes. *Expert Systems with Applications*, 223, 119791. https://doi.org/10.1016/j.eswa.2023.119791. (*Published*).
4. Márquez, R., Weber, R., and Barrales, A. (2024). Detecting disjoint and overlapping communities in temporal node-attributed networks. *Knowledge-Based Systems*. (*Submitted*).

Summary of contributions

The main contributions of this thesis can be summarized as follows:

- We propose two algorithms capable of detecting communities over time by integrating topological structure and node attributes on dynamic attributed networks, addressing a gap in community detection literature where few studies have combined these features.
- Both algorithms adapt to evolving community structure, dynamic topology, and changing node attributes over time, enabling accurate community detection in real-world networks.
- To jointly leverage topological and attribute information in dynamic networks, CoDeDANet employs spectral clustering combined with tensor analysis, while OCoDeDANet utilizes probabilistic non-negative matrix factorization.
- In cases without ground truth, we propose a percentage pairwise comparison method to integrate topological and attribute-based metrics, providing a comprehensive assessment by incorporating both sources of information.
- Results indicate that the advantage of incorporating attributes becomes more pronounced as the community structure derived solely from network topology becomes less distinct.

Through these contributions, this thesis proposes community detection methods that accommodate real-world networks' complex and evolving nature, providing more accurate and insightful community structures.

Thesis structure

In chapter 1, a model for disjoint community detection in dynamic attributed networks with the number of communities as a parameter for the algorithm is described and tested. Next, in chapter 2, an approach for overlapping community detection in dynamic attributed networks with automatic detection of the number of communities is explained, and its performance is compared with state-of-the-art algorithms. Lastly, final remarks about this work are stated, and some recommendations for improvement are mentioned.

Chapter 1

Dynamic community detection including node attributes

1.1. Introduction

A network is a simplified representation of phenomena that occur in the real world, where interactions between objects (nodes) can be represented by connections (links). Examples of such networks include social networks, such as Facebook, contact tracing networks and friendship or business networks; technological networks, such as the internet, power grids and telephone networks; information networks, such as the Worldwide Web, citation networks and collaboration networks; and biological networks, such as food webs and protein–protein interaction networks (Newman, 2018).

These aforementioned networks can be analyzed as static networks but most of them can evolve through time. For example, new contacts will emerge in the contact tracing network, and some people will no longer exist in the network after a period of time; new papers can be cited in the citation networks, or older papers can obtain new citations. The importance of these changes motivates the study of dynamic networks over static networks, since the latter can be considered an oversimplification of the corresponding real-world phenomena.

The representation of real-world systems as networks allows the discovery of useful information in physical, biological and social systems, among others, which could not be found otherwise. The identification of groups in this context, which can be achieved through community detection, is an important task to provide insights for decision-making processes.

Communities (also referred to as modules) are defined as groups of nodes that are more densely connected to each other than to the rest of the network and are also more likely to be linked (Newman, 2006).

Community detection, originally designed in the context of static networks, has been applied in areas such as social networks (Girvan & Newman, 2002; Ferrara et al., 2014; Ozer et al., 2016; Atay et al., 2017), biological systems (Mahmoud et al., 2014; Wu et al., 2014; Taya et al., 2016; Atay et al., 2017), and recommendation systems (Abdrabbah et al., 2014;

Ríos & Videla-Cavieres, 2014; Lalwani et al., 2015), among others. Nevertheless, due to the importance of considering dynamic networks, dynamic community detection emerged (Palla et al., 2007), where time-based approaches are designed to find relevant groups in networks.

Since the structure of social networks may not be sufficient to identify its communities (Bothorel et al., 2015), another feature that can be incorporated into the network data, which is valuable in the context of community detection due to homophily and social influence (Pfeiffer III et al., 2014; Bothorel et al., 2015; Khanam et al., 2022), is the node information. Therefore, the use of attributed networks can be beneficial to accomplish the community detection task.

Furthermore, similar to networks that consider only connections, these attributed networks can also be dynamic, where the nodes and links can appear or disappear, and the attribute values can change over time.

Based on the above discussions, and considering the lack of approaches that use both topology and attribute information, we designed an algorithm for COmmunity DEtection in Dynamic Attributed NETworks (CoDeDANet). The main contributions of this paper are summarized as follows:

- We propose an algorithm that can deal with changes in communities, the structure of the networks, and changes in the attributes of the nodes over time while using information about both the topology and the node features.
- The proposed algorithm integrates the use of spectral clustering to capture the topological information and attributes with the use of tensors, to include the past information.
- The experimental results on different synthetic and real-world networks show that the model outperforms other state-of-the-art community detection algorithms.
- In the absence of ground truth, a percentage pairwise comparison to fuse the topology- and attribute-based metrics is proposed, since both sources of information are considered.

To the best of our knowledge, this is the first approach that combines attributes and topology information in social networks in a dynamic setting, where the number of communities can be introduced as an input.

The paper is organized as follows. Section 1.2 discusses previous works on community detection in static and dynamic networks. Section 1.3 describes a novel approach to detect communities in dynamic attributed networks in a two-phase algorithm. Section 1.4 examines the performance of the proposed model for different synthetic and real-world dynamic attributed networks and compares it with previous works. Finally, in Section 1.5, we conclude our work and provide some insights into possible future work.

1.2. Background

In this paper, we propose a novel method for the dynamic community detection of attributed networks. Section 1.2.1 introduces general state-of-the-art community detection algorithms with a particular emphasis on the novelty of our approach.

Since our paper builds on tensors and spectral clustering for dynamic community detection, a tensor and spectral clustering-based model for community detection that uses only information about the topology in a dynamic network is described in Section 1.2.2.

Finally, since our approach also includes node attributes, in Section 1.2.3, we present a spectral clustering model on a network that includes the attributes (features) of its nodes.

1.2.1. Literature review

A vast number of approaches for community detection have been proposed in social network analysis; see, e.g., Table 1.1, where we classify previous works into four types of algorithms, static without node attributes, dynamic without node attributes, static with node attributes, and dynamic with node attributes.

Table 1.1: Classification of community detection algorithms based on the use of node attribute information and the dynamic aspects of the network.

	Without node attributes'	**With node attributes'**
Static	Blondel et al. (2008) Ma et al. (2014) Cheng et al. (2018) Said et al. (2018) Lu et al. (2020) Shang et al. (2022)	Combe et al. (2015) Cao et al. (2018) Huang et al. (2020) Tang et al. (2020) Ma et al. (2021) Pourabbasi et al. (2021)
Dynamic	Mankad & Michailidis (2013) Wang et al. (2016b) Al-sharoa et al. (2019) Nath et al. (2020) Bahadori et al. (2021) Li et al. (2021) Li et al. (2022)	Bello et al. (2016)

To the extent of our knowledge, only Bello et al. (2016) has proposed an algorithm that includes both dynamic networks, and attributes of the nodes, for community detection. This work was an extension of Blondel et al. (2008) and Combe et al. (2015), and is based on the optimization of a multiobjective function that maximizes modularity to improve the structural quality of partitions, and an attribute similarity function, to include node attribute information. To account for changes in dynamic graphs, Bello et al. (2016) propose a scoring function to measure the degree of change of each vertex between two consecutive time steps. The algorithm then chooses a random number of vertices, with the higher score, to update their communities.

Our approach focuses on addressing the latter and less studied type of algorithm, where community detection is applied on dynamic attributed networks. To accomplish this task, the proposed model of this paper uses spectral clustering and tensors, so these methods are explained thoroughly next.

1.2.2. Tensor and spectral clustering-based dynamic community detection

Spectral clustering is a method that aims to find groups in a similarity graph. Usually, data points are transformed into this by using an ε-neighborhood graph, KNN graph or a

fully connected graph. First, to project the data into a low-dimensional space, a graph Laplacian matrix is obtained. Then, the spectral clustering algorithm solves the trace optimization problem of Eq. (1.1), where $\mathbf{L}_N$ is the normalized Laplacian matrix (Al-sharoa et al., 2019). The solution of this problem is obtained by assigning $\mathbf{U}$ as a matrix containing the K eigenvectors that correspond to the smallest eigenvalues of $\mathbf{L}_N$. Each row of this matrix represents a feature vector of the nodes with size K. Finally, by applying k-means clustering to matrix $\mathbf{U}$, a set of clusters is found (Von Luxburg, 2007).

$$\min_{\mathbf{U}\in\mathbb{R}^{n\times r}} tr\left(\mathbf{U}^\top \mathbf{L}_N \mathbf{U}\right), \quad s.t.\ \mathbf{U}^\top\mathbf{U} = \mathbf{I} \tag{1.1}$$

The main idea of spectral clustering is to find a partition such that the members of the same cluster are similar to each other, and the members of different groups are dissimilar to each other. As shown in Von Luxburg (2007), this can be achieved by solving a minimum cut problem, which aims to minimize the similarity of nodes that belong to different groups. Furthermore, the normalized spectral clustering formulation from Eq. (1.1) can be derived as a relaxation of minimizing the normalized minimum cut problem.

Al-sharoa et al. (2019) also developed a model based on spectral clustering; it considered only topological information but included tensors for handling dynamic networks. In this context, the normalized Laplacian matrix is defined as $\mathbf{L}_N = \mathbf{I} - \mathbf{A}_N$, where $\mathbf{A}_N = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$ is the normalized adjacency matrix, $\mathbf{A}$ is the adjacency matrix, $\mathbf{D}$ is the degree matrix ($D_{\mathrm{ii}} = \sum_j a_{\mathrm{i}j}$) and $\mathbf{I}$ is the identity matrix. Therefore, Eq. (1.1) can be rewritten as shown in Eq. (1.2) and Eq. (1.3) (Al-sharoa et al., 2019).

$$\max_{\mathbf{U}\in\mathbb{R}^{n\times r}} tr\left(\mathbf{U}^\top \mathbf{A}_N \mathbf{U}\right), \quad s.t.\ \mathbf{U}^\top\mathbf{U} = \mathbf{I} \tag{1.2}$$

$$\max_{\mathbf{U}\in\mathbb{R}^{n\times r}} ||\mathbf{U}^\top \mathbf{A}_N \mathbf{U}||_F^2, \quad s.t.\ \mathbf{U}^\top\mathbf{U} = \mathbf{I} \tag{1.3}$$

Different than in Eq. (1.1), the solution to these formulations is to choose $\mathbf{U}$ as a matrix containing the K eigenvectors that correspond to the largest eigenvalues of $\mathbf{A}_N$, instead of the smallest eigenvalues of $\mathbf{L}_N$.

To include past information in the detection process, a weighted average of the normalized adjacency matrices for current and previous time steps can be used (Al-sharoa et al., 2019). This can be written as Eq. (1.4), where L matrices are being used (the current one and $L-1$ previous), and w_l is the weight of the normalized adjacency matrix for time window l.

$$\max_{\mathbf{U},\mathbf{w}} ||\mathbf{U}^\top \sum_{l=1}^{L} w_l \mathbf{A}_N^{(l)} \mathbf{U}||_F^2, \quad s.t.\ \mathbf{U}^\top\mathbf{U} = \mathbf{I}, \mathbf{w} \geq 0, ||\mathbf{w}||_2 = 1 \tag{1.4}$$

This formulation can be rewritten as a Tucker decomposition problem, as shown in Eq. (1.5), where the set of normalized adjacency matrices are represented by a third-order tensor $\boldsymbol{\mathcal{X}}^{(t)} \in \mathbb{R}^{N^{(t)}\times N^{(t)}\times L} = \left[\mathbf{A}_N^{(t-L+1)}, \mathbf{A}_N^{(t-L+2)} \ldots, \mathbf{A}_N^{(t)}\right]$, for each time step $t = 1, \ldots, T$, whose l-th frontal slice is the normalized adjacency matrix $\mathbf{A}_N^{(l)}$ (Al-sharoa et al., 2019).

$$\max_{\mathbf{U},\mathbf{w}} ||\boldsymbol{\mathcal{X}}^{(t)} \times_1 \mathbf{U}^\top \times_2 \mathbf{U}^\top \times_3 \mathbf{w}^\top||_F^2, \quad s.t.\ \mathbf{U}^\top\mathbf{U} = \mathbf{I}, ||\mathbf{w}||_2 = 1 \tag{1.5}$$

The optimization is performed by using higher-order orthogonal iteration (HOOI) (Kolda & Bader, 2009), where the matrix $\mathbf{U}$ or the vector $\mathbf{w}$ is fixed while optimizing the other one in an iterative procedure. One of the first two factor matrices $\mathbf{U}$ of the decomposition gives the input to k-means clustering, and the third factor $\mathbf{w}$, gives the weight of each time step (Al-sharoa et al., 2019).

Since the model of Al-sharoa et al. (2019) does not include node attributes, we next review a model that includes this aspect in a static setting.

1.2.3. Spectral clustering-based community detection using graph distance and node attributes

Tang et al. (2020) proposed a spectral clustering-based method that includes the use of attributes. A Gaussian kernel function is used to measure the structural similarity over the graph distance ($\mathbf{S}_G$) and the attribute similarity over the Euclidean or Hamming distance ($\mathbf{S}_F$). For notation purposes, we denote attributes as $\mathbf{F}$ instead of $\mathbf{X}$, and the resulting matrices as $\mathbf{V}$ instead of $\mathbf{U}$, which is a difference from the original paper.

The Gaussian kernel function to obtain the similarity matrix for the attributes is represented by Eq. (1.6), where α_m $(m = 1, 2, \ldots, M)$ is the weight of the m-th attribute; $\mathbf{D}_F^{(m)} \in \mathbb{R}^{N \times N}$ is the m-th attribute distance matrix between the N nodes, which can be computed by the Euclidean distance or Hamming distance, for numerical or categorical attributes, respectively (Tang et al., 2020); and σ_1 is the width of the local neighborhood relationships, which can be computed as the length of the longest edge in a minimal spanning tree of the graph (Von Luxburg, 2007).

$$s_F(\mathrm{i}, j) = \exp\left\{ -\frac{\sum_{m=1}^{M} \alpha_m \left(\mathrm{d}_F^{(m)}(\mathrm{i}, j) \right)^2}{\sigma_1^2} \right\} \tag{1.6}$$

From each similarity matrix, a normalized matrix is obtained. The K largest eigenvalues in the absolute value of the normalized adjacency matrix $\mathbf{L}_G = \mathbf{D}^{-1/2}\mathbf{S}_G\mathbf{D}^{-1/2}$ are chosen, whose corresponding K eigenvectors (spectrum) are the columns of V_G, i.e., the static structural assignment matrix. The K largest eigenvalues of the normalized attribute matrix $\mathbf{L}_F = \mathbf{D}^{-1/2}\mathbf{S}_F\mathbf{D}^{-1/2}$ are chosen, whose corresponding K eigenvectors are the columns of V_F, i.e., the static attribute assignment matrix. After normalizing both V_G and V_F, these are included in a joint matrix $\mathbf{V} = [\mathbf{V}_G\ \mathbf{V}_F] \in \mathbb{R}^{N \times 2K}$, where k-means clustering is applied to estimate the K clusters (Tang et al., 2020).

The objective of the model of Tang et al. (2020) is to minimize the normalized cut of both the topological structure and the attributes, described by Eq. (1.7) and Eq. (1.8), respectively, where $\mathcal{C}_k$ are the indices for the members of community k.

$$Ncut(G) = \sum_{k=1}^{K} \frac{\sum_{\mathrm{i} \in \mathcal{C}_k} \sum_{j \notin \mathcal{C}_k} s_G(\mathrm{i}, j)}{\sum_{\mathrm{i} \in \mathcal{C}_k} \sum_{j=1}^{n} s_G(\mathrm{i}, j)} \tag{1.7}$$

$$Ncut(F) = \sum_{k=1}^{K} \frac{\sum_{\mathrm{i} \in \mathcal{C}_k} \sum_{j \notin \mathcal{C}_k} s_F(\mathrm{i}, j)}{\sum_{\mathrm{i} \in \mathcal{C}_k} \sum_{j=1}^{n} s_F(\mathrm{i}, j)} \tag{1.8}$$

Tang et al. (2020) propose a procedure for updating the weight of the attributes by simultaneously minimizing the data distance within a group and maximizing the data separation between groups. To achieve this, α_m is updated by Eq. (1.9), using Eq. (1.10), where p represents the iteration number.

$$\alpha_{m,p+1} = \frac{1}{2}\left(\alpha_{m,p} + \Delta\alpha_{m,p}\right) \tag{1.9}$$

$$\Delta\alpha_{m,p} = \frac{f_{m,p}/\mathrm{e}_{m,p}}{\sum_{m=1}^{M} f_{m,p}/\mathrm{e}_{m,p}} \tag{1.10}$$

The overall within-cluster distance for the m-th attribute is represented by $\mathrm{e}_m = \sum_{k=1}^{K} \sum_{n \in \mathcal{C}_k} \mathrm{d}ist(F_{nm}, c_{km})$, where F_{nm} is the value of the m-th attribute for the n-th node, c_{km} is the k-th center of the attribute m and $\mathrm{d}ist(A, B)$ is the distance between A and B. The sum of the distances between the clusters for the m-th attribute is represented by $f_m = \sum_{k=1}^{K} N_k \mathrm{d}ist\left(c_{km}, \bar{F}_m\right)$, where N_k is the number of nodes of the k-th community and $\bar{F}_m = \frac{1}{N}\sum_{n=1}^{N} F_{nm}$ (Tang et al., 2020).

1.3. Proposed framework: CoDeDANet

Based on the developments presented in Section 1.2, we designed a new integrated approach to find communities in dynamic networks that incorporates topology and attribute information in a tensor and spectral clustering-based model. The main contribution of the new framework for COmmunity DEtection in Dynamic Attributed NETworks (CoDeDANet) is the use of past and current information about the network links and node attributes to detect communities. Section 1.3.1 presents the general features of the algorithm, followed by a thorough description in Section 1.3.2.

1.3.1. General overview of CoDeDANet

CoDeDANet works for attributed networks where nodes can appear, disappear, or reappear over time. Edges can also be added or removed over time. Moreover, the values of the node attributes can also change over time. All of these changes can alter the structure of the communities.

The proposed approach works with an unweighted and undirected attributed network with T time steps, $N^{(t)}$ nodes and $K^{(t)}$ communities at time $t = 1, 2, \ldots, T$. Adjacency matrices at time t are represented by $\mathbf{G}^{(t)} \in \mathbb{R}^{N^{(t)} \times N^{(t)}}$, and attribute matrices at time t are represented by $\mathbf{F}^{(t)} \in \mathbb{R}^{N^{(t)} \times M}$, where M is the number of attributes.

CoDeDANet is composed of two phases, which are integrated into one algorithm. In the first phase, the matrices that represent the links and the matrices that represent the nodes' attributes at each time step are obtained by using spectral clustering to optimize the attributes' importance in a setting that joins the nodes' features with a topological structure. In

a second phase, tensors are used to join matrices across time, which are averaged using an optimized weight, whose eigenvalue decomposition allows us to obtain the assignment matrices. Finally, k-means clustering is applied over a join version of the assignment matrices for both sources of information to detect the communities for each time step.

1.3.2. Description of CoDeDANet

The first phase of CoDeDANet is described in Section 1.3.2.1, the second phase in Section 1.3.2.2 and the full algorithm of CoDeDANet in Section 1.3.2.3.

1.3.2.1. Phase 1: computation of normalized matrices

This process can be described using the flowchart of Fig. 1.1, where the inputs are the information about links and attributes. The pseudocode of the process is shown in Algorithm 1, which specifies each step of the first phase of the model.

Algorithm 1: Computation of normalized matrices for CoDeDANet

Input: $\mathbf{G}^{(t)}$, $\mathbf{F}^{(t)}$, $K^{(t)}$, ε, $MaxIter$
Output: $\mathbf{L}_G^{(t)}$, $\mathbf{L}_F^{(t)}$ (For $t = 1$, $\mathbf{V}_G^{(1)}$ and $\mathbf{V}_F^{(1)}$ are also outputs)

Compute $\mathbf{L}_G^{(t)}$, the normalized adjacency matrix of $\mathbf{G}^{(t)}$.
Determine $\mathbf{V}_G^{(t)}$ from the $K^{(t)}$ largest eigenvalues of $\mathbf{L}_G^{(t)}$ in absolute values.
Normalize each row of $\mathbf{V}_G^{(t)}$ to obtain the unit Euclidean norm.
Set $p \leftarrow 1$; $Ncut_{[p]}(\mathbf{F}^{(t)}) \leftarrow 1$
Set $\alpha_{m,[p]}^{(t)} \leftarrow \frac{1}{M}\ \forall\ m = 1, \ldots, M$
Compute $\mathbf{D}_F$ using the Euclidean distance or Hamming distance.
repeat
    Compute $\mathbf{S}_F^{(t)}$ by Eq. (1.6).
    Compute $\mathbf{L}_F^{(t)}$, the normalized attribute matrix of $\mathbf{S}_F^{(t)}$.
    Determine $\mathbf{V}_F^{(t)}$ from the $K^{(t)}$ largest eigenvalues chosen from $\mathbf{L}_F^{(t)}$.
    Normalize each row of $\mathbf{V}_F^{(t)}$ to obtain the unit Euclidean norm.
    Construct $\mathbf{V}_{GF}^{(t)} \leftarrow \left[\mathbf{V}_G^{(t)}\ \mathbf{V}_F^{(t)}\right] \in \mathbb{R}^{N^{(t)} \times 2K^{(t)}}$, normalize each row to have unit Euclidean norm, and perform k-means clustering to obtain the static community assignment $\mathbf{Y}^{(t)}$.
    Compute $Ncut_{[p+1]}(\mathbf{F})$ based on $\mathbf{Y}^{(t)}$ and $\mathbf{S}_F^{(t)}$, as shown in Eq. (1.8).
    Compute $\alpha_{m,[p+1]}^{(t)}\ \forall\ m = 1, \ldots, M$ using Eq. (1.9).
    Update p as $p \leftarrow p + 1$
until $p > MaxIter$ ***or*** $|Ncut_{[p]}(F) - Ncut_{[p-1]}(F)| < \varepsilon$

The top part of Fig. 1.1 shows that the normalized adjacency matrix $\mathbf{L}_G^{(t)} = \mathbf{D}^{-1/2}\mathbf{G}^{(t)}\mathbf{D}^{-1/2}$ is obtained from the topology matrix $\mathbf{G}^{(t)}$, where $\mathbf{D}$ is a diagonal matrix of the nodes' degrees obtained from $\mathbf{G}^{(t)}$. For this case, our approach uses the adjacency matrix $\mathbf{G}^{(t)}$ directly as the structural similarity matrix. The objective of the normalized adjacency matrix is to obtain a representation of the graph where nodes with large degrees are more comparable to nodes with small degrees. The next step in the process is to select the eigenvectors for the

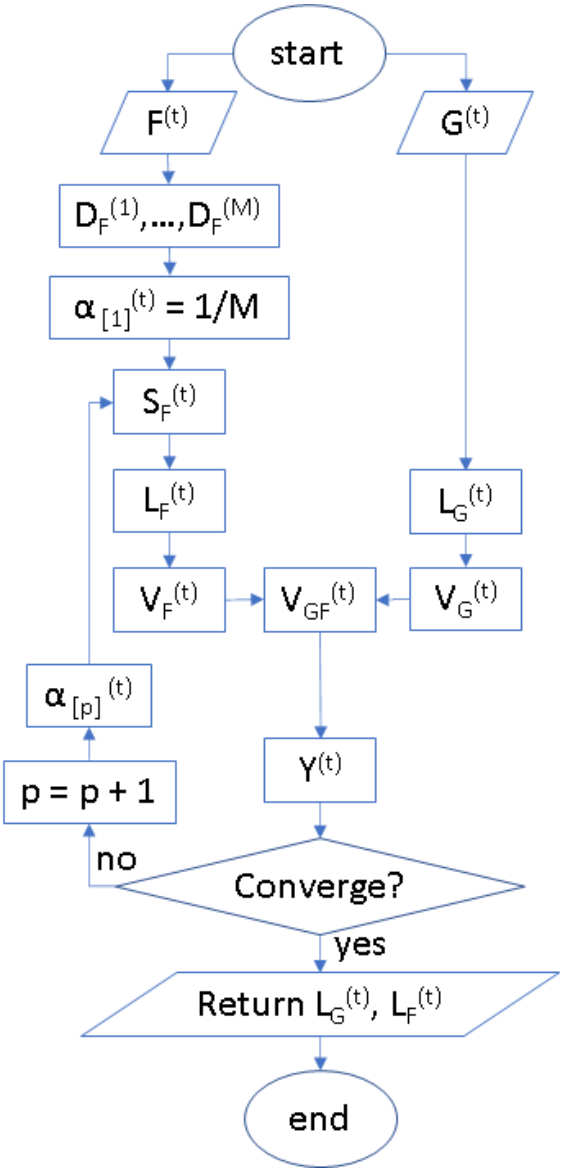


Figure 1.1: Phase 1 of CoDeDANet.

$K^{(t)}$ largest eigenvalues in absolute value from $\mathbf{L}_G^{(t)}$ to project the data into a low-dimensional space representation, which is used to construct the static structural assignment matrix $\mathbf{V}_G^{(t)} \in \mathbb{R}^{N^{(t)} \times K^{(t)}}$. Each row of the resulting matrix is composed of a vector of size $K^{(t)}$, which is the set of features of the corresponding node in the new representation. Since the absolute value of the eigenvalues are considered, the model can be applied to disassortative networks.

The bottom part of Fig. 1.1 shows the processing of the attribute information. Firstly, the attribute matrix $\mathbf{F}^{(t)}$ is used to obtain a distance matrix $\mathbf{D}_F^{(m)}$ for each attribute, where Euclidean distance is applied for numerical attributes, and Hamming distance is applied for categorical attributes. Next, we transform the distance matrices into one attribute similarity matrix, $\mathbf{S_F}^{(t)}$, using a Gaussian kernel function, as described in Eq. (1.6), where each attribute has a weight $\alpha_{m,[p]}^{(t)}$. Afterward, the process follows a similar approach as in the case of the topology. The normalized attribute matrix is computed $\mathbf{L}_F^{(t)} = \mathbf{D}^{-1/2}\mathbf{S_F}^{(t)}\mathbf{D}^{-1/2}$, where $\mathbf{D}$ is

a diagonal matrix of the nodes' degrees obtained from $\mathbf{S_F}^{(t)}$, and the eigenvectors for the K largest eigenvalues are selected from $\mathbf{L}_F^{(t)}$ to construct the static attribute assignment matrix $\mathbf{V}_F^{(t)} \in \mathbb{R}^{N^{(t)} \times K^{(t)}}$. Each row is composed of a vector of size $K^{(t)}$, which is the set of features of the corresponding node in the new representation.

The next step is to form the static joint assignment matrix $\mathbf{V}_{GF}^{(t)} = \left[\mathbf{V}_G^{(t)} \mathbf{V}_F^{(t)}\right]$, using both the static structural assignment matrix and the static attribute assignment matrix. Next, a clustering algorithm must be used to find groups. Given the new representation of the data, which enhances its cluster properties, the k-means clustering algorithm is a good fit at this stage (Von Luxburg, 2007), so it is applied to a normalized version of $\mathbf{V}_{GF}^{(t)}$ to obtain a static community assignment $\mathbf{Y}^{(t)}$. Afterward, using the stopping criteria specified in Algorithm 1, a similar iterative optimization procedure as the one described in Section 1.2.3 is performed, where the attributes' weights α_m are optimized according to the normalized cut measure, with a posterior update of the static attribute similarity matrix $\mathbf{S}_F$.

As shown in Algorithm 1, the outputs used for the next phase are the normalized adjacency matrix $\mathbf{L}_G^{(t)}$ and normalized attribute matrix $\mathbf{L}_F^{(t)}$. For $t = 1$, there is a special case where $\mathbf{V}_G^{(1)}$ and $\mathbf{V}_F^{(1)}$ are also outputs that pass into the next phase.

1.3.2.2. Phase 2: dynamic community discovery

To include past information in the detection process, a weighted average of the normalized matrices for the current and previous time steps can be used, similar to Section 1.2.2.

Fig. 1.2 depicts a flowchart of Phase 2, and Algorithm 2 describes the dynamic process, where $\mathbf{X}_{(3)}^{(t)}$ is the mode-3 matricization of tensor $\boldsymbol{\mathcal{X}}^{(t)}$ (also called unfolding or flattening, where the elements of the tensor are reordered into a matrix along the third dimension (Kolda & Bader, 2009)), the symbol $\bigotimes$ represents the Kronecker product, and L is the size of the time window.

For the graph information, an iterative procedure is performed to update the weight of each time window. As seen in both Fig. 1.2 and Algorithm 2, in the first iteration, a set of inputs are the L normalized adjacency matrices $\mathbf{L}_G^{(t-L+1)}, \mathbf{L}_G^{(t-L+2)}, \ldots, \mathbf{L}_G^{(t)}$, which are used to build the tensor $\boldsymbol{\mathcal{X}}^{(t)}$, to include past information in the clustering process. Another input is the dynamic structural assignment matrix from the previous time step $\mathbf{U}_G^{(t-1)}$, which is used as a starting value for the same matrix at time t, $\mathbf{U}_{G,[0]}^{(t)}$, where the subindex represented as $[p]$ is the iteration number.

With this information, a Tucker decomposition problem similar to the one shown in Eq. (1.5) is formulated, whose solution can be obtained using HOOI (Kolda & Bader, 2009; Al-sharoa et al., 2019). First, the next set of weights for the time windows $\mathbf{w}_{G,[p+1]}$ are computed by a singular value decomposition, as the largest left singular vectors of $\mathbf{X}_{(3)}^{(t)} \left(\mathbf{U}_{G,[p]}^{(t)} \bigotimes \mathbf{U}_{G,[p]}^{(t)}\right)$. Then, these values are used to calculate a weighted average of the normalized adjacency matrices $\bar{\mathbf{L}}_G$, whose eigenvalue decomposition allows us to determine an updated dynamic structural assignment matrix $\mathbf{U}_{G,[p+1]}^{(t)}$. This process is repeated to optimize the weight of each time window l until the stopping criteria of Algorithm 2 are reached. As a result, the low-rank approximations of tensor $\boldsymbol{\mathcal{X}}^{(t)}$ are obtained, where the dynamic

structural assignment matrix for time t, $\mathbf{U}_G^{(t)}$, is an output to be used in the next stage of the model.

Regarding the attribute information, based on a fixed set of weights $\mathbf{w}_F$, a weighted average of the normalized attribute matrices $\bar{\mathbf{L}}_F$ is obtained, whose eigenvalue decomposition allows us to determine the dynamic attribute assignment matrix $\mathbf{U}_F^{(t)}$.

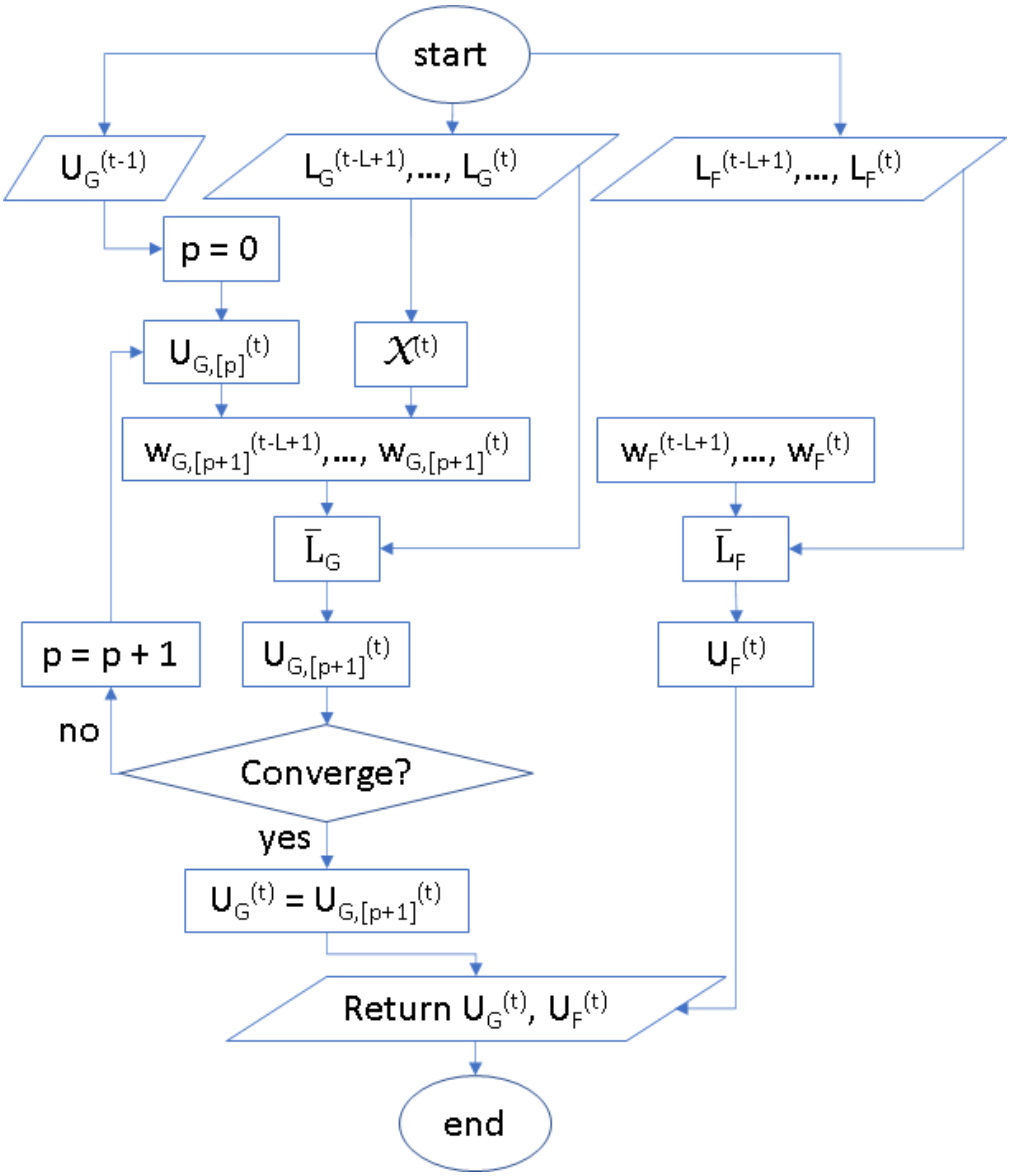


Figure 1.2: Dynamic update of assignment matrices on Phase 2 of CoDeDANet.

Algorithm 2: Dynamic phase of CoDeDANet.

Input: $\mathbf{L}_G^{(t-L+1)}, \mathbf{L}_G^{(t-L+2)} \ldots, \mathbf{L}_G^{(t)}$, $\mathbf{U}_G^{(t-1)}$, $\mathbf{L}_F^{(t-L+1)}, \mathbf{L}_F^{(t-L+2)} \ldots, \mathbf{L}_F^{(t)}$, $\mathbf{w}_F$, ε, $MaxIter$, L

Output: $\mathbf{U}_G^{(t)}$,$\mathbf{U}_F^{(t)}$

Construct the tensor $\boldsymbol{\mathcal{X}}^{(t)} \leftarrow \left[\mathbf{L}_G^{(t-L+1)}, \mathbf{L}_G^{(t-L+2)} \ldots, \mathbf{L}_G^{(t)}\right]$.

Set $p \leftarrow 0$.

Initialize $\mathbf{U}_{G,[p]}^{(t)} \leftarrow \mathbf{U}_G^{(t-1)}$.

repeat

  Compute $\mathbf{w}_{G,[p+1]}$ as the largest left singular vectors of $\mathbf{X}_{(3)}^{(t)} \left(\mathbf{U}_{G,[p]}^{(t)} \bigotimes \mathbf{U}_{G,[p]}^{(t)}\right)$.

  Calculate $\bar{\mathbf{L}}_G \leftarrow \sum_{l=t-L+1}^{t} w_{G,[p+1]}^{(l)} \mathbf{L}_G^{(l)}$.

  Find $\mathbf{U}_{G,[p+1]}^{(t)}$ by eigenvalue decomposition of $\bar{\mathbf{L}}_G$.

  Update p as $p \leftarrow p + 1$.

until $p > MaxIter$ ***or*** $|\mathbf{U}_{G,[p]}^{(t)} - \mathbf{U}_{G,[p-1]}^{(t)}|^2 < \varepsilon$

Assign $\mathbf{U}_G^{(t)} \leftarrow \mathbf{U}_{G,[p]}^{(t)}$.

Calculate $\bar{\mathbf{L}}_F \leftarrow \sum_{l=t-L+1}^{t} w_F^{(l)} \mathbf{L}_F^{(l)}$.

Find $\mathbf{U}_F^{(t)}$ by eigenvalue decomposition of $\bar{\mathbf{L}}_F$.

1.3.2.3. Full algorithm of CoDeDANet

Using the two previous procedures, we define CoDeDANet to detect communities for a dynamic network using information about both the graph and nodes' attributes. A sketch of the full process is depicted in Fig. 1.3, where $L = 2$ for simplicity. The pseudocode is shown in Algorithm 3.

First, Phase 1 is used to obtain the normalized matrices and assignment matrices for the first time step. Then, an iterative procedure for each time step is used, starting at $t = 2$, from the creation of normalized matrices through Phase 1, then obtaining the dynamic structural and attribute assignment matrix, $\mathbf{U}_G^{(t)}$ and $\mathbf{U}_F^{(t)}$, respectively, by Phase 2, and finally applying k-means clustering on the normalized version of the dynamic joint assignment matrix $\mathbf{U}_{GF}^{(t)} = \left[\mathbf{U}_G^{(t)}\ \mathbf{U}_F^{(t)}\right]$, to find the communities $\mathbf{Z}^{(t)}$ at each time step.

In the next section, different datasets are used to evaluate the performance of CoDeDANet in comparison with other state-of-the-art community detection algorithms.

1.4. Experimental results and evaluation

To evaluate the model we measured its performance on synthetic attributed networks with ground truth and two real-world attributed social networks without ground truth[1].

[1]The datasets used in this paper are available at `https://data.mendeley.com/datasets/fkz6mbpr2z`

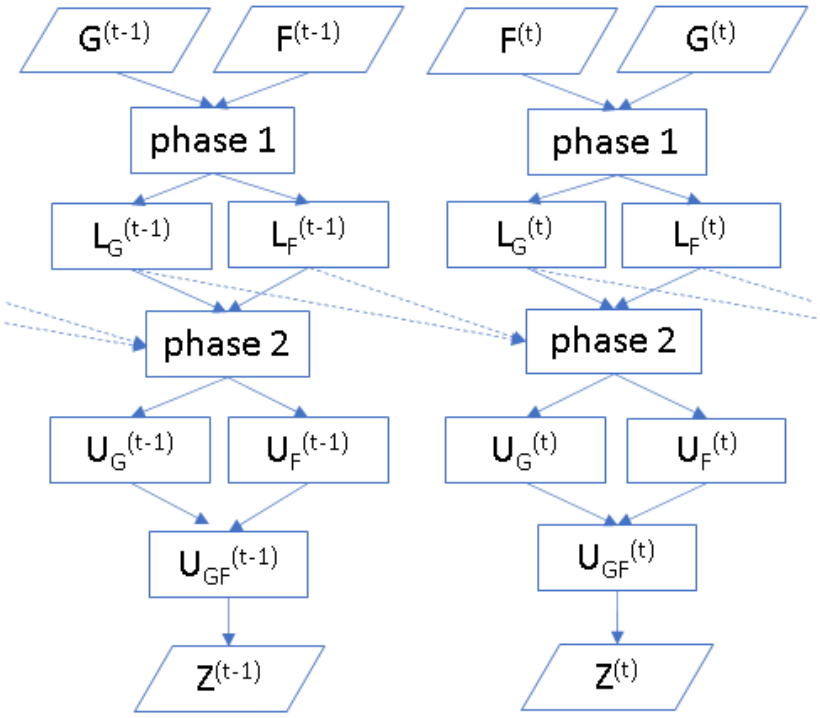


Figure 1.3: Sketch of CoDeDANet.

Algorithm 3: Full algorithm of CoDeDANet.

Input: $\mathbf{G}^{(1)}, \ldots, \mathbf{G}^{(T)}$, $\mathbf{F}^{(1)}, \ldots, \mathbf{F}^{(T)}$, $K^{(1)}, \ldots, K^{(T)}$, ε, $MaxIter$, L

Output: Clustering labels of each node for each time step $\mathbf{Z}^{(t)} \in \mathbb{R}^{N^{(t)} \times K^{(t)}}$ $\forall\, t = 1, \ldots, T$

Determine $\mathbf{L}_G^{(1)}$, $\mathbf{L}_F^{(1)}$, $\mathbf{V}_G^{(1)}$, $\mathbf{V}_F^{(1)} \leftarrow$ Algorithm 1 ($\mathbf{G}^{(1)}$, $\mathbf{F}^{(1)}$, $K^{(1)}$, ε, $MaxIter$)

Assign $\mathbf{U}_G^{(1)} \leftarrow \mathbf{V}_G^{(1)}$, $\mathbf{U}_F^{(1)} \leftarrow \mathbf{V}_F^{(1)}$

for *t=2 to T* **do**

  Determine $\mathbf{L}_G^{(t)}$, $\mathbf{L}_F^{(t)} \leftarrow$ Algorithm 1 ($\mathbf{G}^{(t)}$, $\mathbf{F}^{(t)}$, $K^{(t)}$, ε, $MaxIter$)

  Determine $\mathbf{U}_G^{(t)}$, $\mathbf{U}_F^{(t)} \leftarrow$ Algorithm 2 ($\mathbf{L}_G^{(t-L+1)}, \mathbf{L}_G^{(t-L+2)} \ldots, \mathbf{L}_G^{(t)}$, $\mathbf{U}_G^{(t-1)}$, $\mathbf{L}_F^{(t-L+1)}, \mathbf{L}_F^{(t-L+2)} \ldots, \mathbf{L}_F^{(t)}$, $\mathbf{w}_F$, ε, $MaxIter$, L)

  Select the first $K^{(t)}$ columns from $\mathbf{U}_G^{(t)}$ and $\mathbf{U}_F^{(t)}$ to construct $\left[\mathbf{U}_G^{(t)}\ \mathbf{U}_F^{(t)}\right] \in \mathbb{R}^{N^{(t)} \times 2K^{(t)}}$, normalize each row to have unit Euclidean norm, and apply k-means clustering to obtain $\mathbf{Z}^{(t)}$.

CoDeDANet is compared with the approach of Bello et al. (2016) (DALouvain), which makes use of both topology and attributes in a modularity-based approach[2]. Furthermore, the model is compared to a static approach with attributes from Tang et al. (2020) (SwA) and a dynamic approach without attributes from Al-sharoa et al. (2019) (DwoA).

The results are structured in the following way: firstly, an overview of the metrics used to

[2]DALouvain implementation is available at `https://bitbucket.org/harenbergsd/dynamic-attributed-louvain/src/master/`

evaluate the performance of the algorithm is presented in Section 1.4.1. Next, in Section 1.4.2, various disjoint and overlapping attributed networks are described and the performance in the different algorithm is addressed. Lastly, Section 1.4.3 provides a brief summary of the results obtained.

1.4.1. Metrics for evaluation of performance in community detection

The performance of community detection algorithms can be measured by different metrics according to the features of the algorithm and the nature of the network. When ground-truth information regarding the community structure is available, one of the most commonly used metrics is normalized mutual information (NMI). In the absence of the ground-truth information, modularity and density are some of the metrics used to capture the presence of community structures for community detection (Chakraborty et al., 2017; Chunaev, 2020). If the network has categorical attributed data describing the nodes, the cohesiveness of the network according to this information can be measured using entropy (Chunaev, 2020). When the attributed data are numerical, Calinski–Harabasz or Davies–Bouldin metrics (Arbelaitz et al., 2013) can be used.

Normalized mutual information is a measure to determine the quality of a clustering. To explain this metric, a confusion matrix $\boldsymbol{N}$ is defined, where the rows are the class labels obtained from the ground truth, and the columns are the communities obtained by a community detection algorithm. An element N_{ij} from this matrix represents the number of nodes with class label i that appear in the found community j. The NMI is defined as Eq. (1.11), where A is the ground-truth partition, $\mathcal{C}$ is the found partition, $n_{\mathcal{A}}$ is the number of ground-truth communities, $n_{\mathcal{C}}$ is the number of found communities and $N_{i.}$ is the sum over row i, which defines the number of nodes of class label i, and $N_{.j}$ is the sum over column j, which defines the number of nodes of the found community j (Danon et al., 2005).

$$NMI(\mathcal{A},\mathcal{C}) = \frac{-2\sum_{i=1}^{n_{\mathcal{A}}}\sum_{j=1}^{n_{\mathcal{C}}} N_{ij}\log\left(N_{ij}N/\left(N_{i.}N_{.j}\right)\right)}{\sum_{i=1}^{n_{\mathcal{A}}} N_{i.}\log\left(N_{i.}/N\right)+\sum_{j=1}^{n_{\mathcal{C}}} N_{.j}\log\left(N_{.j}/N\right)} \tag{1.11}$$

The NMI equals 1 if the partitions are identical and 0 if the partitions are independent.

The *density* of partition $\mathcal{C}$ is defined as shown in Eq. (1.12), where m_k is the number of internal edges of community k, m is the number of edges of the graph and K is the number of communities (Zhou et al., 2009; Dang & Viennet, 2012).

$$Density(\mathcal{C}) = \sum_{k=1}^{K}\frac{m_k}{m} \tag{1.12}$$

The density measures the proportion of community internal links according to the total number of links in the graph. The higher the density is, the better the partition (Dang & Viennet, 2012).

The entropy measures the similarity of the nodes within a community k, according to the values of a categorical attribute m, as shown in Eq. (1.13), where X_m is a vector of values for

attribute m, $\mathcal{C}_k$ represent the nodes that belong to community k, n_m is the number of values for attribute m, and p_{mkn} is the percentage of nodes in community k with a value X_{mn} for the attribute m (Zhou et al., 2009; Dang & Viennet, 2012).

$$entropy(X_m, \mathcal{C}_k) = -\sum_{n=1}^{n_m} p_{mkn} \log(p_{mkn}) \tag{1.13}$$

The *entropy* of partition $\mathcal{C}$ allows us to measure the performance of an algorithm according to the similarity of the values of the categorical attributes within each community and is defined as shown in Eq. (1.14), where N_k is the number of nodes that belong to community k. The lower the entropy is, the more homogeneous the nodes inside each community according to the values of its attributes (Zhou et al., 2009; Dang & Viennet, 2012).

$$Entropy(\mathcal{C}) = \sum_{m=1}^{M} \frac{\alpha_m}{\sum_{p=1}^{M} \alpha_m} \sum_{k=1}^{K} \frac{N_k}{N} \cdot entropy(X_m, \mathcal{C}_k) \tag{1.14}$$

1.4.2. Experiments on synthetic and real-world networks

To evaluate the performance of CoDeDANet, four synthetic attributed networks with ground-truth information were created. When using DALouvain as the community detection algorithm, the attributes of these networks were discretized to be able to use the available code.

For each parameter setting to generate data of every synthetic network, 10 different instances were created, each one with a different random seed. The number of communities for the synthetic networks was set with the same value as the ground truth.

To show how our method performs if the ground truth is not available, we used two real-world attributed social networks: a COVID-19 contact tracing and a crime network.

As parameters of our model, we fixed the size of the time window as $L = 2$, $\varepsilon = 0{,}00001$ and $MaxIter = 20$. The k-means algorithm was run with 20 different random seed centers.

According to initial tests, the weight of the past information for the attributes was fixed to 0, i.e., only attribute information for the current time step was used in the experiments.

1.4.2.1. Synthetic network 1

Networks

We use synthetic network 1 to show the creation of a community. For these tests, synthetic dynamic networks similar to the ones from Sheikholeslami & Giannakis (2018) were used. The graphs were built with 200 nodes and 20 snapshots. The network starts with the nodes divided into two communities, each one with 100 nodes. Afterwards, a third community appears. Two independent datasets were built, to show the results when more changes occur. For Dataset 1, 40 % of all nodes from the entire network migrate to a new community. For Dataset 2, 80 % of all nodes migrate to a new community. The time of the change for each

node is selected according to a normal distribution $\mathcal{N}(10, 2)$. For each pair of nodes, a link is created according to a stochastic block model, where nodes within the same community are connected with probability 0.3, and nodes from different communities are connected with probability 0.1.

For the generation of the attributes, we propose four cases using a univariate or multivariate normal distribution with a standard deviation of 0.1.

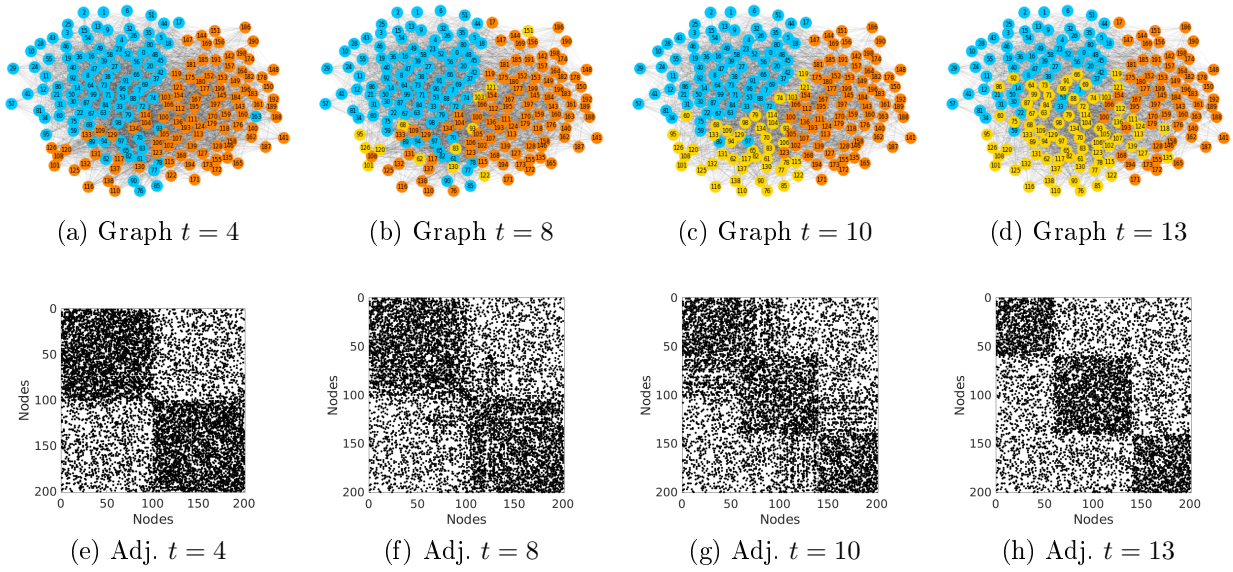
In Case 1, a three-variate normal distribution is used to generate three attributes, with means $\mu_1 = [1, 0, 0]$, $\mu_2 = [0, 0, 0]$, $\mu_3 = [-1, 0, 0]$, for the nodes belonging to Groups 1, 2, and 3, respectively. Hence, only the first attribute contributes to identifying the groups.

In Case 2, we propose to use only one attribute, with means $\mu_1 = 1$, $\mu_2 = 0$ and $\mu_3 = -1$ for nodes belonging to Groups 1, 2 and 3, respectively.

In Case 3, there are three irrelevant random attributes.

In Case 4, there is one irrelevant random attribute.

The graphs, and an adjacency matrix representation of some selected timestamps, for an instance of Dataset 1 and Case 1, are shown in Fig. 1.4. In the upper part of Fig. 1.4, the communities found by CoDeDANet are distinguished by colors. In the lower part of Fig. 1.4, a black dot indicates the existence of a link between the respective node on the horizontal axis and the one on the vertical axis.



(a) Graph $t = 4$ (b) Graph $t = 8$ (c) Graph $t = 10$ (d) Graph $t = 13$

(e) Adj. $t = 4$ (f) Adj. $t = 8$ (g) Adj. $t = 10$ (h) Adj. $t = 13$

Figure 1.4: An instance of graphs and adjacency matrices, for Dataset 1, Case 1, of Synthetic network 1.

The evolution of the network shows that some nodes start to change their connections and attributes, initiating the birth of a new community. Attribute 1 over time supports the same community evolution, as shown in Fig. 1.5, where the lower the value is, the darker the color.

As can be seen in Fig. 1.4, some nodes start to change their connections, initiating the birth of a new community. Attribute 1 supports the same community evolution over time, as shown in Fig. 1.5, where the lower the value is, the darker the color.

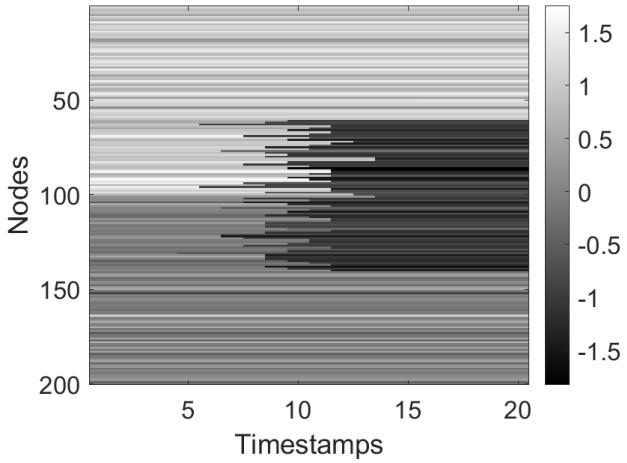


Figure 1.5: Attribute 1 over time, for Dataset 1, of Synthetic network 1.

Results

The performance of the models for Dataset 1 of Synthetic network 1 is shown in Fig. 1.6. For all models, a performance decay begins around time step 5, starting to partially recover around timestamp 8 in most cases.

For the Dataset 1 of Synthetic network 1, when there is one relevant attribute (Fig. 1.6a and Fig. 1.6b), CoDeDANet and SwA perform best, followed by DwoA, since the attributes help to recover the performance faster.

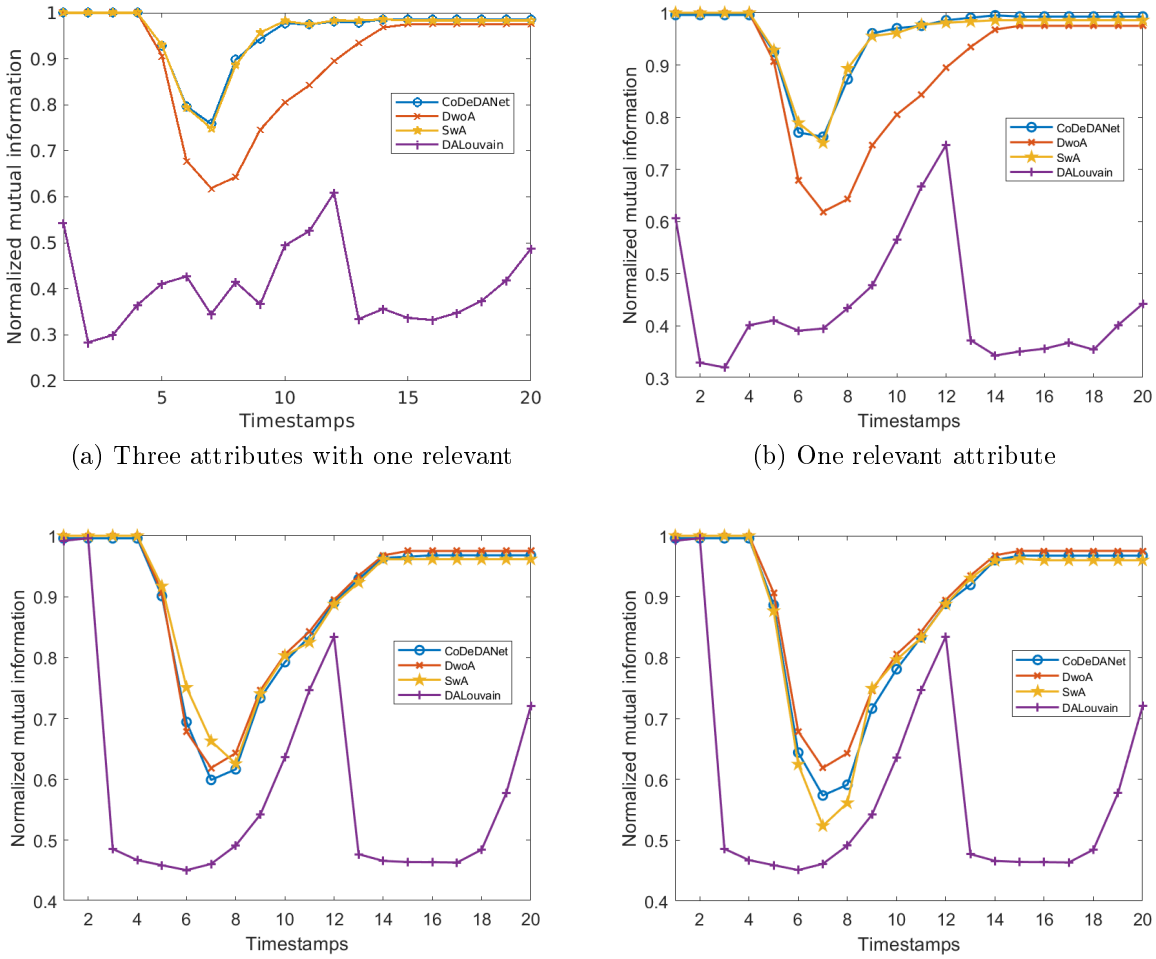
When attributes are random (Fig. 1.6c and Fig. 1.6d), despite learning the weights of the attributes, the influence of the irrelevant attributes is strong enough to worsen the performance. The results are similar among CoDeDANet, DwoA and SwA, with the same average NMI when there are three irrelevant random attributes and less than a 1.1 % difference on average between them when there is just one irrelevant attribute.

Fig. 1.7 shows the models' performance for Dataset 2 of Synthetic network 1. Similar to Dataset 1, on the cases that include relevant attributes (Fig. 1.7a and Fig. 1.7b), there is a decay in performance starting around time step 3, finally stabilizing at approximately 0.54 for CoDeDANet, 0.5 for SwA and 0.43 for DwoA. For Dataset 2 of Synthetic network 1, CoDeDANet outperforms the other models. When the attributes are random (Fig. 1.7c and Fig. 1.7d), DwoA performs best, followed by CoDeDANet and SwA.

For both datasets, DALouvain performs worst, producing only one community in most cases.

1.4.2.2. Synthetic network 2

Networks



(a) Three attributes with one relevant

(b) One relevant attribute

(c) Three irrelevant attributes

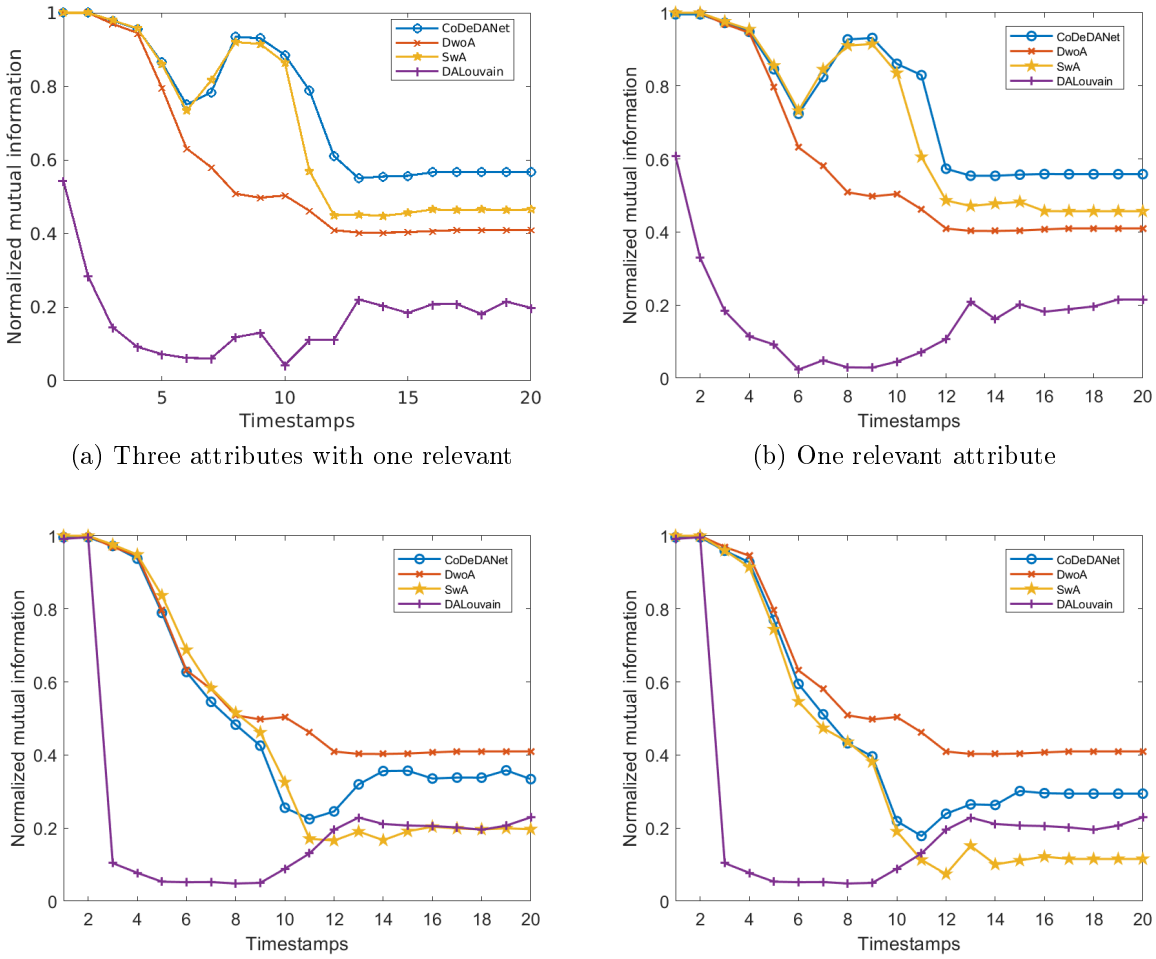
(d) One irrelevant attribute

Figure 1.6: Performance for Dataset 1 of Synthetic network 1 measured by NMI, where 40 % of the nodes migrate to a new community.

With these networks we show how communities grow and shrink. We created data similar to those used by Tang et al. (2020) but generated 10 timestamps with communities that grow and shrink. We built 3 datasets, each one with 3 groups of $n_1 = 80$, $n_2 = 90$ and $n_3 = 100$ nodes. The network structure is generated by a degree-corrected stochastic block model. The link probability is defined by $z \cdot (\theta_\mathrm{i}) \cdot (\theta_j)$, where $z \in [0, 1]$ is the original link probability of connecting the nodes i and j of a certain group, and $\theta_\mathrm{i}, \theta_j$ are the degree-corrected parameters. We set $\theta_\mathrm{i} = 1$ for 95 % of the nodes and $\theta_\mathrm{i} = 3$ for the remaining nodes.

Three types of attributed networks were proposed, with different original link probabilities, to assess strongly assortative structures, weakly assortative structures, and disassortative structures, using the parameter values proposed in Tang et al. (2020), which are shown in Table 1.2, where $p_1 \in [0{,}21, 0{,}25]$, $p_2 = p_1 - 0{,}01$, $p_3 = p_1 - 0{,}02$, $q \in [0{,}1, 0{,}14]$, and $r \in [0{,}19, 0{,}23]$.

Similar to the previous networks, a three-variate normal distribution is used to generate the 3 attributes, with means $\mu_1 = [1{,}5, 0, 0]$, $\mu_2 = [0, 0, 0]$, $\mu_3 = [-1{,}5, 0, 0]$, for nodes belonging



(a) Three attributes with one relevant

(b) One relevant attribute

(c) Three irrelevant attributes

(d) One irrelevant attribute

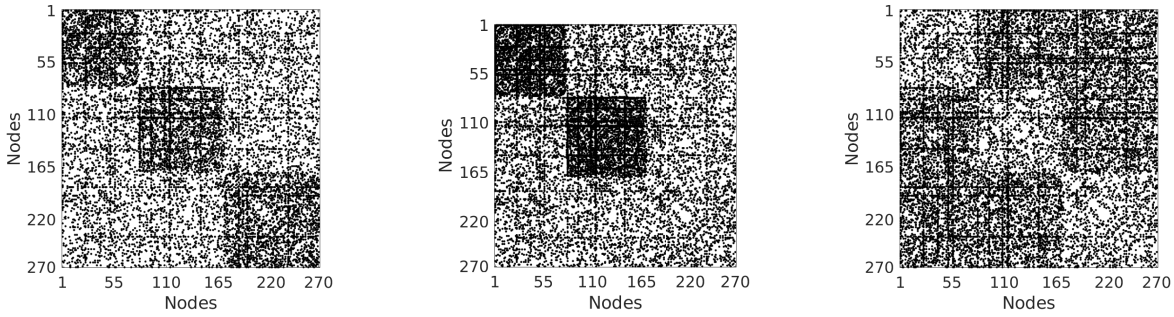
Figure 1.7: Performance for Dataset 2 of Synthetic network 1 measured by NMI, where 80 % of the nodes migrate to a new community.

Table 1.2: Original link probabilities to create datasets for Synthetic network 2.

	Within groups link probabilities			Between groups link probabilities		
	1	2	3	1-2	1-3	2-3
Strongly assortative	p_1	p_2	p_3	0.08	0.08	0.08
Weakly assortative	0.3	0.3	0.1	q	0.09	0.09
Disassortative	0.1	0.1	0.1	r	r	r

to Groups 1, 2 and 3, respectively.

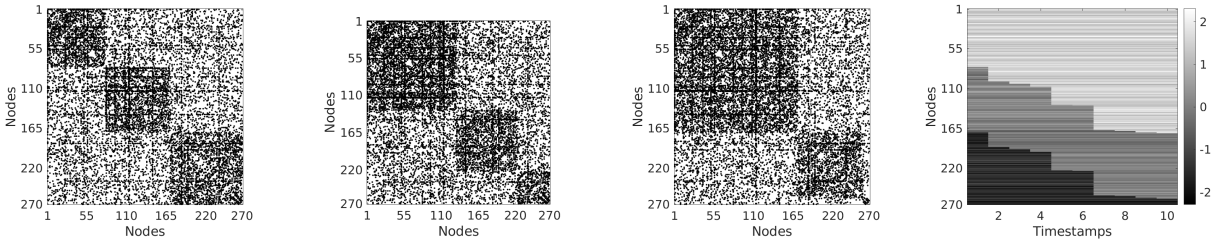
For each type of network, an instance of the adjacency matrices for time step 1 is shown in Fig. 1.8, with parameters $p_1 = 0{,}21$, $q = 0{,}1$ and $r = 0{,}19$.



(a) Strongly assortative (b) Weakly assortative (c) Disassortative

Figure 1.8: An instance of the adjacency matrices for the 3 types of Synthetic network 2 at $t = 1$.

At each of the 10 time steps, changes in links and attributes force the first community to grow, the second community to change its members but remain constant in size, and the third community to shrink, as shown in Fig. 1.9, for an instance of the strongly assortative network with $p = 0{,}21$.



(a) Adj. $t = 1$ (b) Adj. $t = 5$ (c) Adj. $t = 10$ (d) Attr. 1

Figure 1.9: An instance of adjacency matrices and attribute 1 over time for the strongly assortative network with $p = 0{,}21$.

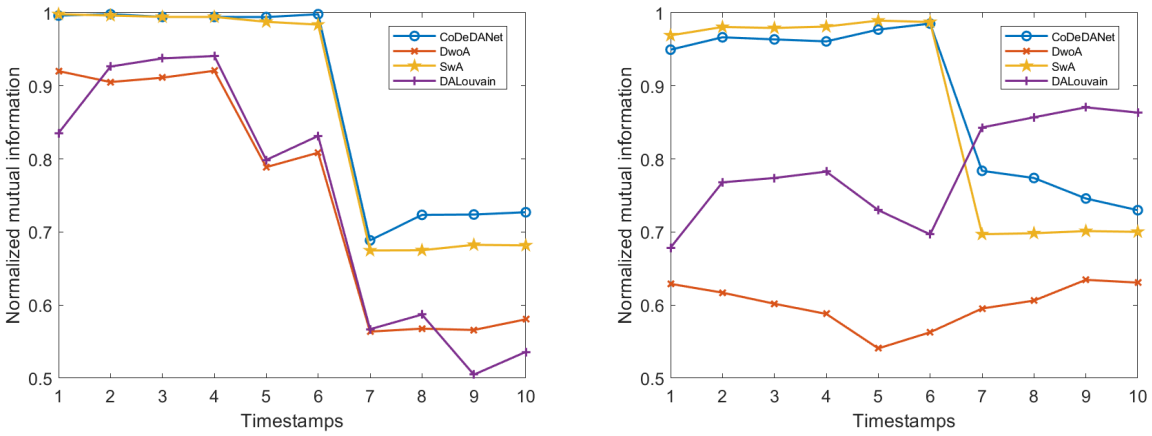
Results

The performance of the models for $p_1 = 0{,}21$, $q = 0{,}1$ and $r = 0{,}19$ is shown in Fig. 1.10. For the strongly and weakly assortative networks, CoDeDANet performs best on average, followed by SwA. In the case of disassortative networks, SwA performs best, followed by CoDeDANet. Both algorithms experience a decay of performance at time step 7, which coincides with the greater change of attribute values across all timestamps.

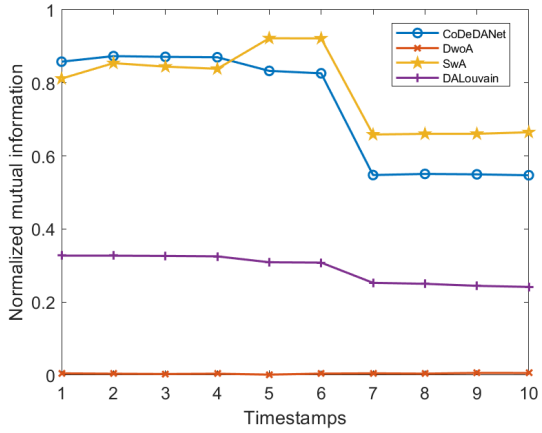
1.4.2.3. Synthetic network 3

Networks

With these networks we show various kinds of changes, such as creation and deletion of communities, as well as increase and decrease of the link density among nodes inside and outside the communities. We design synthetic dynamic networks similarly to Al-sharoa et al. (2019), with the addition of attributes. Four different datasets, each one consisting of attributed networks of 100 nodes and 60 time points were generated. Intra- and inter-cluster



(a) Strongly assortative (b) Weakly assortative



(c) Disassortative

Figure 1.10: Performance measured by NMI for Synthetic network 2.

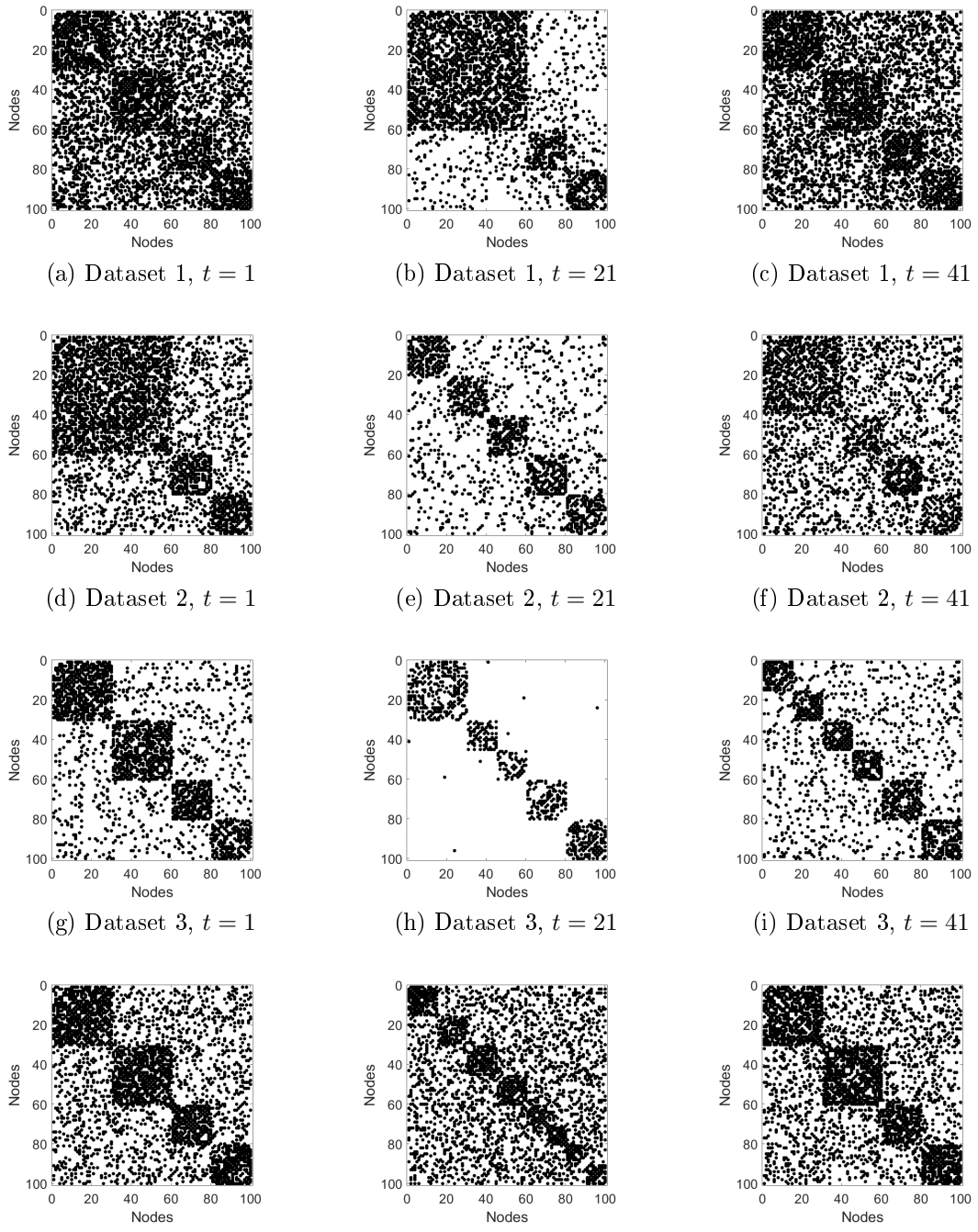
edges are selected from a truncated Gaussian distribution in the range $[0, 1]$. Changes in the structure of the networks are generated between time $t = 20$ to $t = 21$ and $t = 40$ to $t = 41$. Further information about the structure of the datasets' parameters are provided in Al-sharoa et al. (2019).

Figure 1.11 shows the topology of an instance of the synthetic networks. The evolution of the networks shows the birth and death of communities over time. Additionally, the density of the internal edges inside a community and the external edges between nodes of different communities can change.

Attributes were generated similarly to Synthetic network 1, but in this case, the number of groups, and hence the number of means to generate the values of an attribute, varies between three and eight to strengthen the communities when the attributes are relevant.

Results

The results on a network with three attributes where one is relevant are shown in Fig. 1.12.



(a) Dataset 1, $t = 1$ (b) Dataset 1, $t = 21$ (c) Dataset 1, $t = 41$

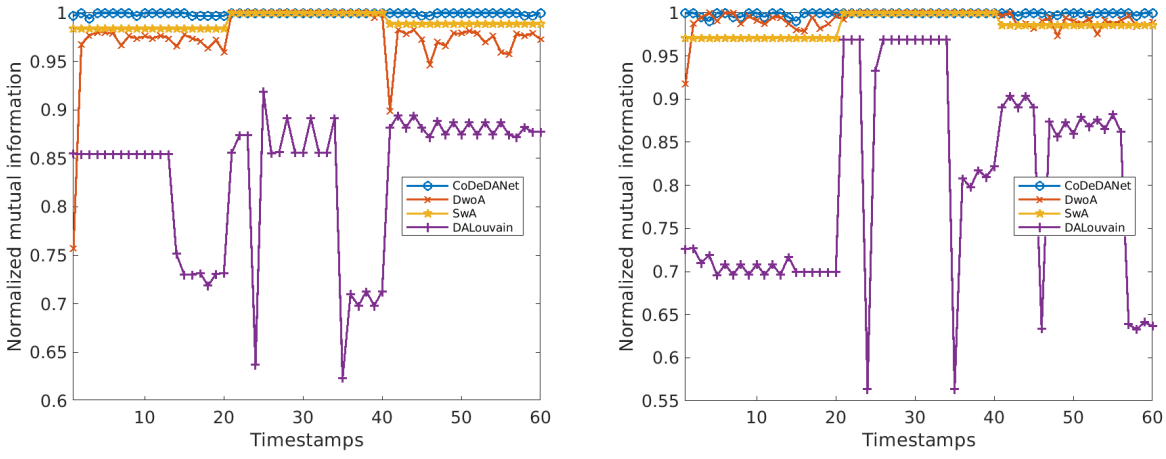
(d) Dataset 2, $t = 1$ (e) Dataset 2, $t = 21$ (f) Dataset 2, $t = 41$

(g) Dataset 3, $t = 1$ (h) Dataset 3, $t = 21$ (i) Dataset 3, $t = 41$

(j) Dataset 4, $t = 1$ (k) Dataset 4, $t = 21$ (l) Dataset 4, $t = 41$

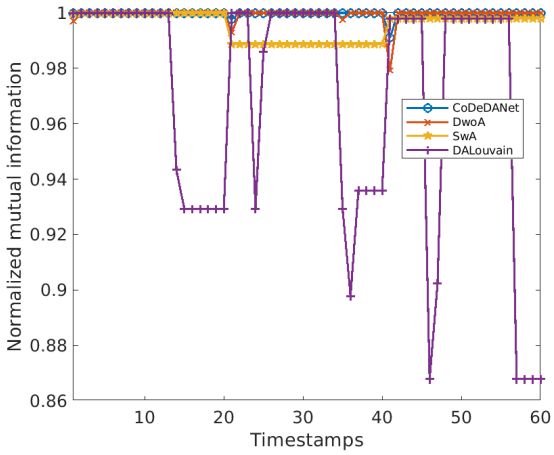
Figure 1.11: Adjacency matrices over time for Synthetic network 3.

A greater stability of performance across changes is exhibited by the models that include attributes. Since the network for Dataset 1 has more intercommunity edges between $t = 1$ and $t = 20$ and $t = 41$ to $t = 60$, as shown by Fig. 1.11a and Fig. 1.11c (more black spots on the nondiagonal at those time steps), respectively, the performance of DwoA decays at those ranges of time steps, as shown in Fig. 1.12a. In contrast, the use of attributes on CoDeDANet and SwA avoids a significant loss of performance in those time periods. DALouvain has the most unstable performance on this set of synthetic networks.

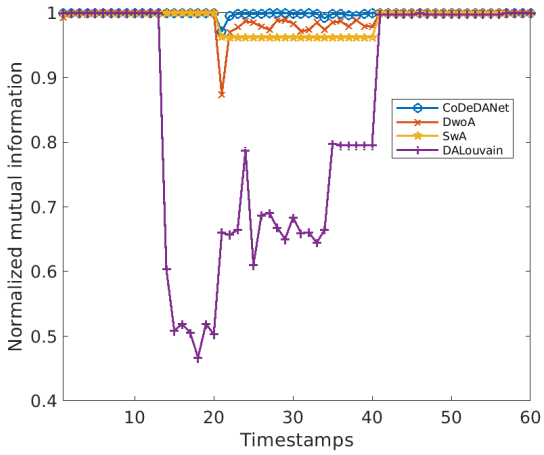


(a) Dataset 1

(b) Dataset 2



(c) Dataset 3



(d) Dataset 4

Figure 1.12: Performance measured by NMI for Synthetic network 3.

1.4.2.4. Synthetic network 4

We use the synthetic benchmark DANCer (Largeron et al., 2017) to create attributed graphs with undirected edges that can change over time, where nodes are grouped into densely connected sets, relatively homogeneous according to the attributes.

In this benchmark, each network is generated by building the first graph with properties such as preferential attachment, small world, or homophily, which is then modified with micro and macro operations. The former adds or removes nodes and edges and updates the attribute values, while the latter splits communities, merges communities, and migrates members to another existing community or a new one (Largeron et al., 2017).

Networks

The general parameters for all the networks created with this benchmark are shown in Table 1.3, including the description for each parameter. The number of nodes, the number

of communities and the number of edges increase over time, ending with a maximum value of 4814 nodes, 15 communities and 21908 edges among the built networks. Twelve different datasets were built using the specific parameters shown in Table 1.4.

Table 1.3: Parameter values used for benchmark DANCer. The description is shown as defined by Largeron et al. (2017).

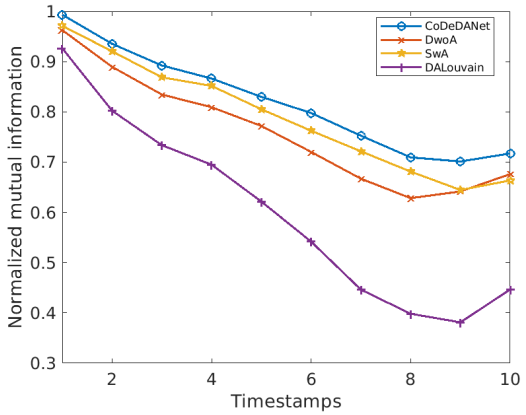
Parameter	Values	Description
N_{init}	1000	Number of nodes at $t = 0$
K_{init}	10	Number of communities at $t = 0$
$NbRep$	10	Maximum number of representatives of each community
P_{rc}	0	A threshold to decide whether a new vertex joins a randomly selected community or not
M	2	Number of numerical attributes
Dev_{i}	1	Standard deviations of the ith attributes generated using centered normal distributions
E_{wth}^{max}	10	Maximum number of edges connecting a new vertex to vertices in its community
E_{btw}^{max}	5	Maximum number of edges connecting a new vertex to vertices in a different community
MTE	5000	Minimum number of total edges
$R_{updateAttrs}$	0.3	Ratio defining the number of attributes updated
P_{micro}	0.5	A threshold to select if the micro dynamic updates are performed or not
R_{awe}	0.3	Ratio defining the number of within edges inserted
R_{rbe}	0.3	Ratio defining the number of between edges removed
R_{abe}	0.5, 0.9	Ratio defining the number of between edges inserted
R_{rwe}	0.5, 0.9	Ratio defining the number of within edges removed
R_{av}	0.2, 0.3, 0.8	Ratio defining the number of vertices inserted
R_{rv}	0.3, 0, 0.9	Ratio defining the number of vertices removed
P_{merge}	0.3, 0.8	Probability to perform the merge operation
P_{split}	0.3, 0.8	Probability to perform the split operation
$P_{migrate}$	0.3, 0.8	Probability to perform the migrate vertices operation
T	10, 10, 5	Number of graphs generated

Table 1.4: Specific parameters for networks defined using the benchmark DANCer.

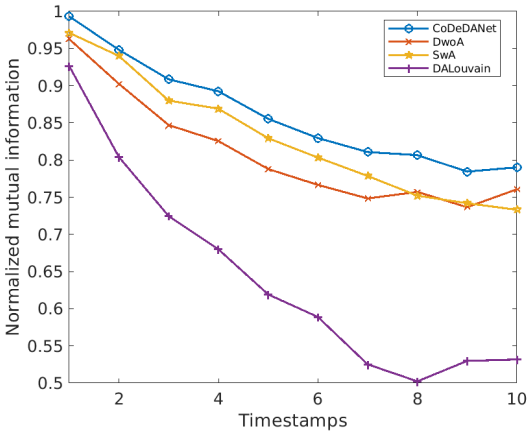
Dataset	R_{rwe}, R_{abe}	R_{av}	R_{rv}	P_{merge}, P_{split}, $P_{migrate}$	T
1	0.5	0.2	0.3	0.3	10
2	0.5	0.2	0.3	0.8	10
3	0.9	0.2	0.3	0.3	10
4	0.9	0.2	0.3	0.8	10
5	0.5	0.3	0	0.3	10
6	0.5	0.3	0	0.8	10
7	0.9	0.3	0	0.3	10
8	0.9	0.3	0	0.8	10
9	0.5	0.8	0.9	0.3	5
10	0.5	0.8	0.9	0.8	5
11	0.9	0.8	0.9	0.3	5
12	0.9	0.8	0.9	0.8	5

Results

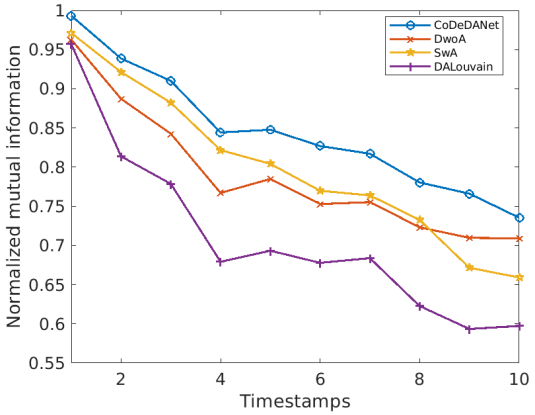
The results for 6 of the 12 built networks are shown in Fig. 1.13. On the available code, DALouvain cannot remove nodes because there is no tracking of the node ID, so the results for DALouvain are obtained in a static manner when $R_{rv} \neq 0$. Over the tested networks, CoDeDANet performs best, followed by SwA and DwA. When the macro operations on communities increase, there is no clear effect on the performance of the model, since Fig. 1.13b displays a better performance than Fig. 1.13a; Fig. 1.13d shows a worse performance than Fig. 1.13c; and Fig. 1.13f shows a similar performance compared to Fig. 1.13e.



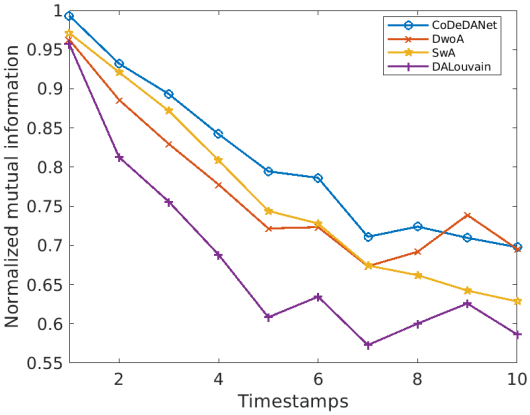
(a) Results for network 3



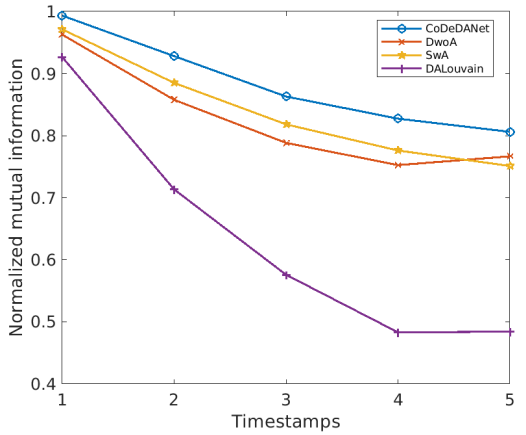
(b) Results for network 4



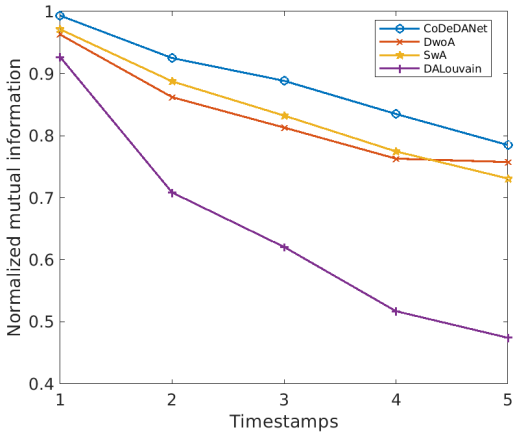
(c) Results for network 7



(d) Results for network 8



(e) Results for network 11



(f) Results for network 12

Figure 1.13: Performance measured by NMI for benchmark DANCer.

1.4.2.5. COVID-19 contact tracing network

The data for the COVID-19 contact tracing network were collected from September 1st, 2020, to January 26, 2021, southeast of the Metropolitan Region of Santiago, Chile.

Since the ground truth for this real-world social network was not available, we had to determine the number of communities for each time step. For CoDeDANet, the number of communities was determined based on the performance using a combination of one topological measure (density) and one attribute measure (entropy).

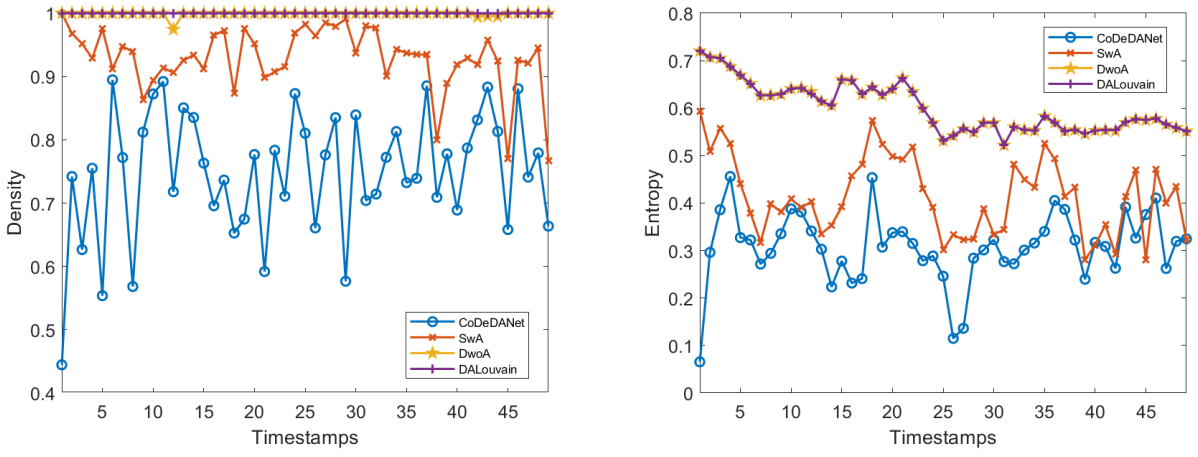
For the case of the algorithms SwA and DwoA, this was defined using the number of communities obtained by DALouvain.

Network

The attributed network consists of 49 time steps, each summarizing a 3-day period, with a maximum of 305 nodes and 214 edges at time step 43 and a minimum of 42 nodes and 27 edges at time step 1. A node represents an individual, and an edge represents that there was close contact between two individuals. Each individual has several attributes, such as age, gender and risk of contact, among others, and most of them have missing data. After preprocessing the data, the final network without missing data used in this paper had only the attribute *risk of contact* selected for community detection. This attribute is set by a professional and refers to the risk of developing dangerous symptoms, which can be $low = 0$, $medium = 1$ or $high = 2$.

Results

The performance of CoDeDANet and the other models is shown in figure 1.14. The best results for density (the greater the better) are obtained by DALouvain and DwoA, followed by SwA and CoDeDANet. The best results for entropy (the lesser the better) are shown by CoDeDANet, followed by SwA, DALouvain and DwoA. This means that CoDeDANet gives more importance to the attributes and DALouvain to the topology. Since there is no clear advantage to an algorithm when including both metrics, no direct conclusions can be made. Section 1.4.3 proposes an approach to address this drawback.



(a) Density (b) Entropy

Figure 1.14: Performance for COVID-19 network.

To take a closer look to the behavior of CoDeDANet, we choose the time step ($t = 43$) having the greatest number of nodes to display its communities (305 nodes). This is

shown in Fig. 1.15 for communities with 5 nodes or more. The color indicates the community membership.

We review some examples where the influence of both the topology and the attributes can be checked. Nodes 221, 222, 224, 227, 230, 247 and 267 are connected, assigned to the same community, and all of them have an attribute value of 2 with the exception of 221 and 227. Nodes 66, 69, 70, 131, 132, 137, 138, 184, 202, 203, 268, 269, 283, 285, 290, 291, 293 and 294, belong to the same community because each one has an attribute value of 2; nevertheless, node 284, which is connected to one these nodes (283), is not a member of the same community because it has an attribute value of 1.

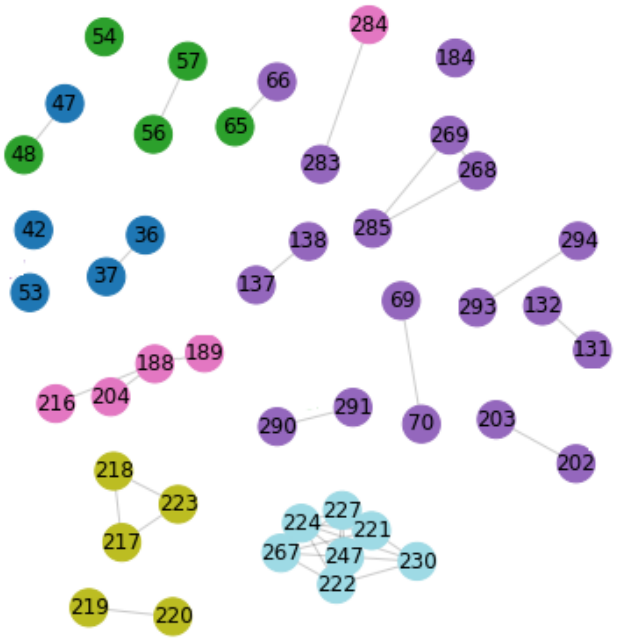


Figure 1.15: Communities obtained by CoDeDANet at $t = 43$ for the COVID-19 network.

1.4.2.6. Crime network

The crime network relates individuals who committed property crimes from January 1st, 2019, to December 31st, 2021, in Chile. Examples of property crimes are *robbery in an uninhabited place*, *robbery with violence*, and *homicide in fight*.

The number of communities for all algorithms used in this network was defined using the value obtained by DALouvain.

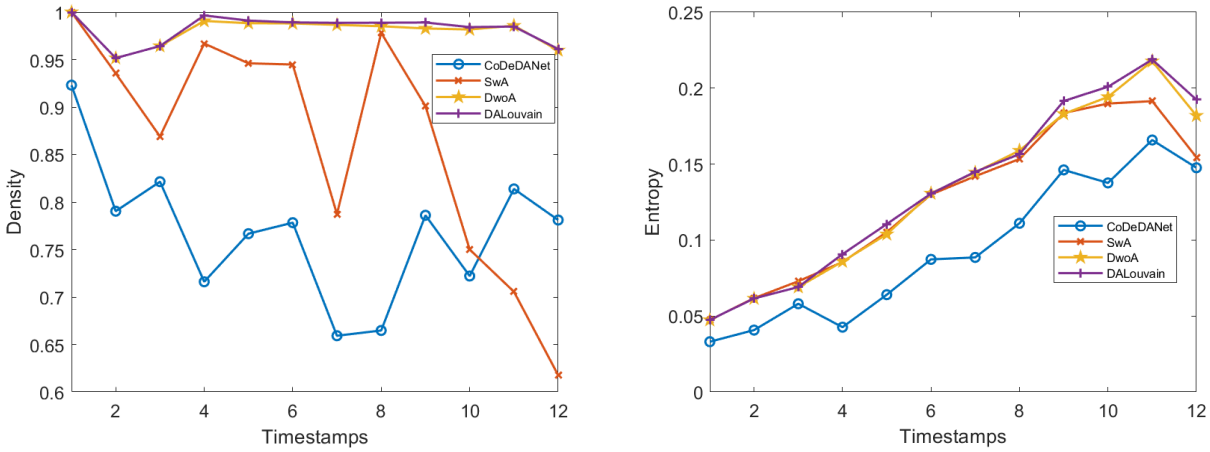
Network

The attributed network is composed of 12 time steps, which are determined based on the date of the crime. Each time step adds crimes from a period of 3 months. The network starts with 38 nodes and 39 edges, and ends with 419 nodes and 2087 edges. For this network there is no deletion of vertices. Each node represents a person who committed a crime and an edge indicates that two individuals were charged in the same crime. The network has a set of

attributes $\boldsymbol{F}^{(t)}$, where a value $F_{nm}^{(t)}$ of 1 indicates that the subject n has committed the type of crime m up to time step t, where m specifies one of the 26 types of crimes on this network.

Results

The performance of CoDeDANet and the other models is shown in figure 1.16. The ranking of the algorithms according to each metric is similar to the one obtained in the COVID-19 dataset. These results are summarized in Section 1.4.3.



(a) Density (b) Entropy

Figure 1.16: Performance for the crime network.

To have some insights on the results of CoDeDANet, Fig. 1.17 shows the three largest communities of the crime network for time step 12.

The third largest community, composed of 21 nodes and shown on the left upper part of Fig. 1.17, has a dense connection between most of its nodes, but it also has three nodes that are not connected to the rest of this group, but are assigned to this community due to their attribute values.

The second largest community, composed of 22 nodes and shown on the left lower part of Fig. 1.17, has less connections between its nodes than the other two communities in the figure. Seven of the nodes are not connected to the rest of this group, but are assigned to this community due to their attribute values.

The largest community, composed of 39 nodes and shown to the right of Fig. 1.17, has a denser connection of its nodes, and all these nodes are connected.

1.4.3. Summary of results

1.4.3.1. Synthetic networks

Previous results showed the performance over time for each dataset of the synthetic attributed networks. Table 1.5 presents a comparison of the different algorithms according to NMI, averaging the results of all time steps and all seeds for each model.

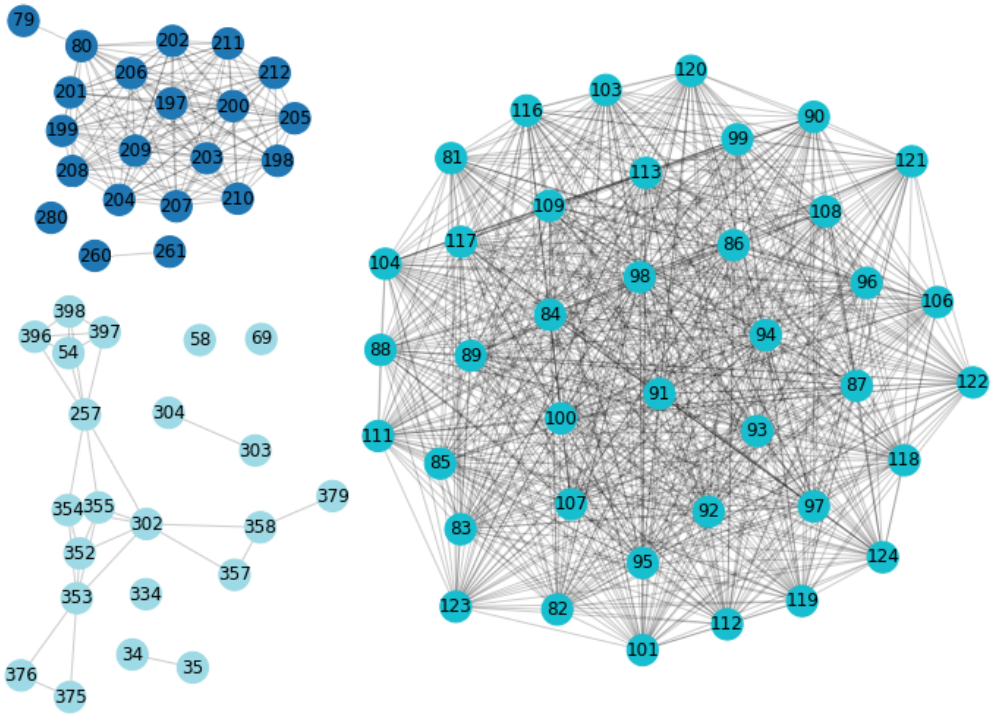


Figure 1.17: Communities obtained by CoDeDANet at $t = 12$ for the crime network.

According to the average NMI, CoDeDANet outperformed the other models that were selected for comparison in most cases.

Averaging over all test networks, CoDeDANet has the highest NMI with 0.854, followed by SwA with 0.834, DwoA with 0.783, and DALouvain with 0.609. Hence, our proposed model outperformed DALouvain in NMI, which also considers both sources of information (attribute values and topology) by 40 %. Furthermore, it also surpasses both cases that only consider the attributes or the dynamic aspect separately, with improvements of 2.4 % and 9.1 %, respectively.

1.4.3.2. Real-world social networks

For both networks, since there is no clear advantage of an algorithm when including both metrics, an option to compare the algorithms is to normalize each metric and compute the average. Nonetheless, the scales of the aforementioned metrics differ vastly, so we decided to compare the algorithms on a one *vs.* one approach. This pairwise comparison consists of measuring the percentage difference in density and entropy and then computing the average.

Table 1.6 and Table 1.7 show the results for the COVID-19 network and the crime network, respectively. In both cases, CoDeDANet is the best among the compared algorithms. For the COVID-19 network, CoDeDANet outperforms SwA by 3.4 %, DwoA by 11.9 % and DALouvain by 11.8 %. For the crime network, CoDeDANet outperforms SwA by 7.3 %, DwoA by 4.8 % and DALouvain by 4.4 %. As expected, DwoA and DALouvain perform better on the topology metric, followed by SwA and CoDeDANet. The best result according to the attribute metric is CoDeDANet, followed by SwA, DALouvain and DwoA. This means that CoDeDANet is building its communities mainly by their attributes in comparison with the

Table 1.5: Performance on synthetic networks according to NMI. The rows comprise the built networks, which number refers to the corresponding figure of results in the paper.

Network	CoDeDANet	SwA	DwoA	DALouvain
Syn. net. 1, Dataset 1, Case 1, Fig. 1.6a	*0.96*	*0.96*	0.89	0.4
Syn. net. 1, Dataset 1, Case 2, Fig. 1.6b	*0.96*	*0.96*	0.89	0.44
Syn. net. 1, Dataset 1, Case 3, Fig. 1.6c	*0.89*	*0.89*	*0.89*	0.58
Syn. net. 1, Dataset 1, Case 4, Fig. 1.6d	0.88	0.88	*0.89*	0.58
Syn. net. 1, Dataset 2, Case 1, Fig. 1.7a	*0.75*	0.69	0.58	0.17
Syn. net. 1, Dataset 2, Case 2, Fig. 1.7b	*0.74*	0.69	0.58	0.16
Syn. net. 1, Dataset 2, Case 3, Fig. 1.7c	0.51	0.46	*0.58*	0.23
Syn. net. 1, Dataset 2, Case 4, Fig. 1.7d	0.48	0.39	*0.58*	0.23
Syn. net. 2, Dataset 1, Fig. 1.10a	*0.88*	0.87	0.75	0.75
Syn. net. 2, Dataset 2, Fig. 1.10b	*0.88*	0.87	0.6	0.79
Syn. net. 2, Dataset 3, Fig. 1.10c	0.73	*0.78*	0	0.29
Syn. net. 3, Dataset 1, Case 1, Fig. 1.12a	*1*	0.99	0.98	0.83
Syn. net. 3, Dataset 2, Case 1, Fig. 1.12b	*1*	0.99	0.99	0.8
Syn. net. 3, Dataset 3, Case 1, Fig. 1.12c	*1*	*1*	*1*	0.97
Syn. net. 3, Dataset 4, Case 1, Fig. 1.12d	*1*	0.99	0.99	0.85
Syn. net. 4, Dataset 1	*0.88*	0.86	0.84	0.73
Syn. net. 4, Dataset 2	*0.88*	0.86	0.84	0.71
Syn. net. 4, Dataset 3, Fig. 1.13a	*0.82*	0.79	0.76	0.6
Syn. net. 4, Dataset 4, Fig. 1.13b	*0.86*	0.83	0.81	0.64
Syn. net. 4, Dataset 5	*0.86*	0.84	0.81	0.76
Syn. net. 4, Dataset 6	*0.87*	0.86	0.84	0.78
Syn. net. 4, Dataset 7, Fig. 1.13c	*0.85*	0.8	0.79	0.71
Syn. net. 4, Dataset 8, Fig. 1.13d	*0.81*	0.77	0.77	0.68
Syn. net. 4, Dataset 9	*0.9*	0.87	0.86	0.71
Syn. net. 4, Dataset 10	*0.89*	0.86	0.86	0.72
Syn. net. 4, Dataset 11, Fig. 1.13e	*0.88*	0.84	0.83	0.64
Syn. net. 4, Dataset 12, Fig. 1.13f	*0.89*	0.84	0.83	0.65
Average	*0.854*	0.834	0.783	0.609

other algorithms.

Table 1.6: Performance on the COVID-19 network according to density and entropy.

	Density	Entropy	Average
CoDeDANet	0.75	0.31	-
SwA	0.93	0.42	-
CoDeDANet *vs* SwA (%)	-19.4	26.2	3.4
DwoA	1	0.60	-
CoDeDANet *vs* DwA (%)	-25.1	49	11.9
DALouvain	1	0.60	-
CoDeDANet *vs* DALouvain (%)	-25.2	48.9	11.8

Table 1.7: Performance on the crime network according to density and entropy.

	Density	Entropy	Average
CoDeDANet	0.77	0.09	-
SwA	0.87	0.13	-
CoDeDANet *vs* SwA (%)	-11.3	26	7.3
DwoA	0.96	0.18	-
CoDeDANet *vs* DwA (%)	-21.6	29.2	4.8
DALouvain	0.96	0.19	-
CoDeDANet *vs* DALouvain (%)	-21.8	30.5	4.4

1.5. Concluding remarks and future work

Real-world processes demand the use of algorithms that are able to capture their evolving nature and complexity. Nonetheless, dynamic attributed networks have been studied mostly in a static context including the attributes, or in a dynamic setting that uses only information about the links of the graphs. Considering these limitations, in this paper, we have proposed CoDeDANet, an algorithm for community detection in dynamic attributed networks that is able to process changes in the structure of the graph and the attributes of its nodes over time. This method integrates the use of spectral clustering to capture the information of topology and attributes with the use of tensors, to include past information.

The model was tested on different synthetic networks and two real-world networks and was compared with other state-of-the-art community detection algorithms. To compare the different models, normalized mutual information for networks with ground truths, and density and entropy for networks without ground truths, were used. The experimental results showed that the proposed model outperformed the algorithms chosen for comparison.

Our model was able to show the benefits of using topology alongside attributes on evolving networks to improve the community detection process. Nevertheless, there are opportunities for further improvement. For example, it would be useful to include a mechanism for automatically determining the number of communities. Additionally, since in our approach the weights of the attribute matrices at each time step are previously fixed, the model would benefit from a procedure to automatically update these weights.

Another aspect to improve the model, is finding ways to deal with irrelevant attributes. If all the attributes used for any reason are not truly relevant to the topology, for example, random attributes, the model would consider the information anyway. In this sense, for the model to only include attributes when they are relevant, an improvement that can be included is to compare the $Ncut$ measure, using the attributes instead of not using the attributes, on the first phase and, according to that, decide if the second phase includes the attributes.

According to the tests we performed, using the Gaussian kernel function on the adjacency matrix caused a decrease in performance on average; nonetheless, other transformations to measure similarity can be used, such as using the modularity matrix of the original adjacency

matrix. Finally, the size of the networks that can be processed by the proposed model is another aspect that could be improved, which could be addressed by updating only the eigenvalues that are affected by changes and not compute the full matrix on each time step; a procedure of this type must be designed.

Chapter 2

Detecting disjoint and overlapping communities in temporal node-attributed networks

2.1. Introduction

Social network analysis provides insights into people's behavior and their interactions. One of the approaches to address this task is community detection. In its most common use, community detection is applied on static networks, based on the links between nodes, to find disjoint or non-overlapping groups, called *partitions* (Blondel et al., 2008; Lu et al., 2020).

Nevertheless, other aspects can be included in the process, such as node features (Qin et al., 2018; Huang et al., 2020; Ma et al., 2021) and evolution of the network over time (Yu et al., 2019; Nath et al., 2020; Gao et al., 2020). Furthermore, algorithms can be designed to find overlapping groups, called *covers* (Nath et al., 2020; Gao et al., 2020; Shang et al., 2022).

We propose a model to detect disjoint and overlapping communities on dynamic attributed networks, using a probabilistic non-negative matrix factorization approach. Our model is solved by using multiplicative update rules (Lee & Seung, 1999, 2001).

Considering that to the best of our knowledge, there are no approaches for overlapping community detection that use both topology and attribute information on dynamic networks, thoroughly tested on multiple networks, we designed an algorithm for Overlapping COmmunity DEtection in Dynamic Attributed NETworks (OCoDeDANet). The main contributions of this paper are summarized as follows:

- We propose an algorithm that infers disjoint and overlapping communities given the links' and nodes' information of a dynamic attributed network.
- The proposed method uses automatic relevance determination to detect the number of communities on the dynamic attributed networks.
- The experimental results on different synthetic attributed networks and one real-world

attributed network show that the model outperforms other state-of-the-art disjoint community detection algorithms.

- The results indicate that the benefits of using attributes increase as the community structure, according to the network's topology, becomes more confusing.

Section 2.2 categorizes and discusses previous work on community detection, according to the nature of the networks and the capabilities of the algorithms, with emphasis on non-negative matrix factorization. Section 2.3 introduces our novel approach to detect overlapping communities on dynamic node-attributed networks, using non-negative matrix factorization and Bayesian techniques to automatically detect the number of communities. Section 2.4 shows the results of the proposed algorithm and previous works for different synthetic and real-world dynamic node-attributed networks. Finally, in Section 2.5, we conclude our work and propose further developments.

2.2. Background

In recent years, several community detection methods have been proposed (Jin et al., 2023). Some reviews address the state-of-the-art algorithms for disjoint community detection on static networks (Bedi & Sharma, 2016; Fortunato & Hric, 2016), overlapping community detection (Xie et al., 2013; Amelio & Pizzuti, 2014), dynamic community detection (Rossetti & Cazabet, 2018; Cazabet & Rossetti, 2023), and community detection on node-attributed networks (Chunaev, 2020). Other surveys have focused on categorizing the algorithms according to the methodology they used (Jin et al., 2023). Regarding the approach used in this paper, a review of non-negative matrix factorization for community detection has also been published (He et al., 2022).

We categorize several previous approaches for community detection in Table 2.1. They are classified as static or dynamic, with or without node features, with a disjoint or overlapping classification, and with a predefined number of communities or an automatic finding of the number of communities.

As shown in Table 1.1, to the best of our knowledge, a method that puts these characteristics all together and thoroughly tested on multiple networks, has yet to be developed. Preliminary results of a simplified and different version of our model were shown in Márquez et al. (2019).

Next, we focus on selected overlapping community detection methods our model builds on, with a special interest in generative models for dynamic networks with node information.

Psorakis et al. (2011) use a model based on Bayesian non-negative matrix factorization as proposed by Tan & Févotte (2009), for overlapping community detection on static networks. The observed variable is considered as a non-negative count of interactions in a weighted undirected network with adjacency matrix V. The expected number of interactions between node i and j, v_{ij} (Poisson distributed), is the result of mutual participation in the same communities, calculated by multiplying matrices W and H (latent variables). The number of communities K is obtained by hyperparameter β, that will be large if contributions of W and H are small, i.e., do not contribute to interactions. The objective is to maximize the

Table 2.1: Classification of community detection algorithms according to network dynamics, availability of nodes' attributes, capability of identifying overlapping nodes, and determination of number of communities

Approaches	Dynamic	Attributes	Overlapping	#Comms
Combe et al. (2015)		✓		✓
Ma et al. (2021)		✓		✓
Márquez & Weber (2023)	✓	✓		
Bello et al. (2016)	✓	✓		✓
Psorakis et al. (2011)			✓	✓
Coscia et al. (2012)			✓	✓
Shang et al. (2022)			✓	✓
Niu et al. (2023)			✓	✓
Yang et al. (2013)		✓	✓	
Qin et al. (2018)		✓	✓	
Huang et al. (2018)		✓	✓	✓
Reihanian et al. (2023)		✓	✓	✓
Ma & Dong (2017)	✓		✓	
Yu et al. (2019)	✓		✓	
Wang et al. (2016a)	✓		✓	✓
Rossetti et al. (2017)	✓		✓	✓
Nath et al. (2020)	✓		✓	✓
Gao et al. (2020)	✓		✓	✓

model posterior given the observations, which also can turn into a minimization of a loss function. Each value H_{ik} denotes the degree of participation of node i to community k, which can be normalized to express soft membership distribution. The model is solved by using multiplicative update rules (Lee & Seung, 1999, 2001).

Afsariardchi (2012) developed a method for overlapping community detection on temporal networks, extending the work from Psorakis et al. (2011). They add the temporal dimension to the model and include a linking term between previous and current communities assignment matrices in the joint probability, modeled by a Gamma distribution.

Huang et al. (2018) designed an algorithm for overlapping community detection with node information, based on Bayesian non-negative matrix factorization. Their work is based on Psorakis et al. (2011), adding the node features matrix to the graphical model, using a Poisson distribution to model the expectation of this matrix.

2.3. The proposed model

In this section, we propose our approach OCoDeNANet, to find overlapping communities on dynamic attributed networks. After introducing the necessary notation, we develop the mathematical model and show the iterative algorithm to solve it. At the end of this section, we present the full algorithm of OCoDeNANet.

2.3.1. Notations

The topology of a dynamic attributed network is defined as $G = \{\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_T\}$, where T is the number of snapshots, $\mathbf{V}_t$ is the adjacency matrix of the undirected and unweighted graph at timestamp t, and $v_{ij,t} = 1$ if node i and node j are connected by a link at time step t and equals 0, otherwise.

The node information of a dynamic attributed network is defined as $F = \{\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_T\}$, $\mathbf{F}_t$ is a binary feature matrix at timestamp t, and $f_{mj,t} = 1$ if node j has the attribute m at time step t and equals 0, otherwise.

The number of nodes at time step t is N_t, the number of features at time step t is $M_t = M$, which is constant over time, and the maximum possible number of communities at time step t is $K_t = K$, which is also constant over time.

When processing matrices, $\frac{\mathbf{X}}{\mathbf{Y}}$ denotes element-wise division of $\mathbf{X}$ and $\mathbf{Y}$. $\mathbf{X} \circ \mathbf{Y}$ denotes the Hadamard product (element-wise multiplication). The matrix $\mathbf{1}_{A \times B}$ denotes a matrix of dimension $A \times B$ where each element is 1.

A letter κ added to an equation represents all constants that are not relevant to the optimization process.

2.3.2. Mathematical model

The model shown in Fig. 2.1 represents the generation structure, where a counting process for $\mathbf{V}_t$ is represented by latent factors $\mathbf{W}_t$ and $\mathbf{H}_t$ (Psorakis et al., 2011), another counting process for $\mathbf{F}_t$ is represented by latent factors $\mathbf{G}_t$ and $\mathbf{H}_t$ (Huang et al., 2018), and scale hyper-parameters $\boldsymbol{\beta}_t$ and fixed hyperhyperparameters $a_{k,t} = a$ and $b_{k,t} = b$ are imposed over these matrices. The latent factors are updated not only by considering the information of $\mathbf{V}_t$ and $\mathbf{F}_t$, but also knowledge of communities obtained from $\mathbf{W}_{t-1}$, $\mathbf{H}_{t-1}$, and $\mathbf{G}_{t-1}$ (Afsariardchi, 2012).

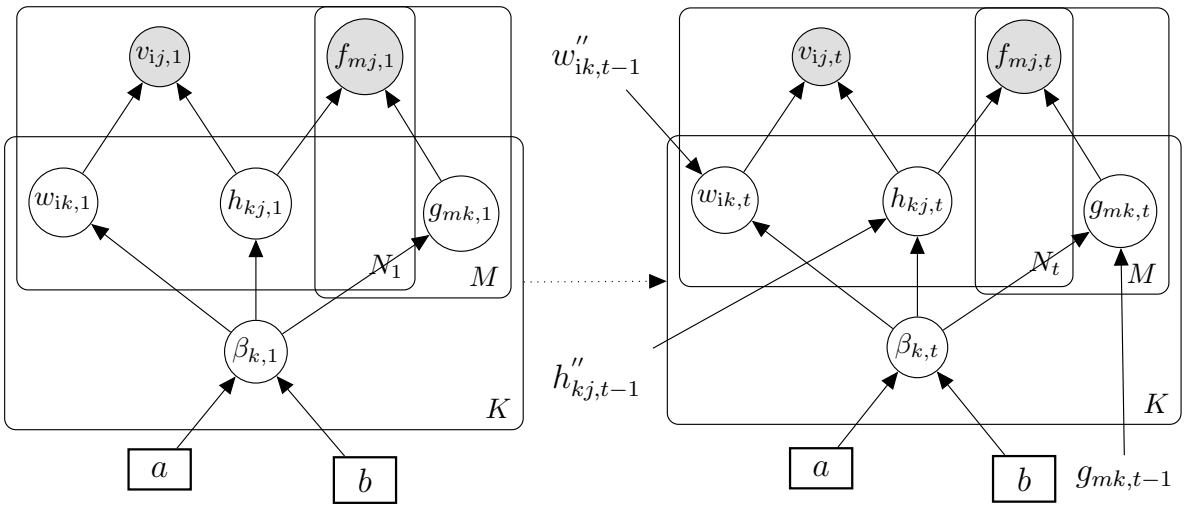


Figure 2.1: Dynamic attributed non-negative matrix factorization (left for $t = 1$, right for $t = 2, \ldots, T$)

The model objective is to find matrices $\hat{\mathbf{V}}_t$, $\forall t = 1, 2, \ldots, T$, and $\hat{\mathbf{F}}_t$, $\forall t = 1, 2, \ldots, T$, that represent the original adjacency matrices $\mathbf{V}_t \in \{0,1\}^{N_t \times N_t}$ and $\mathbf{F}_t \in \{0,1\}^{M \times N_t}$, respectively. The expectation adjacency network is represented by $\hat{\mathbf{V}}_t = \mathbf{W}_t\mathbf{H}_t$, with non-negative matrices $\mathbf{W}_t \in \mathbb{R}_{\geq 0}^{N_t \times K}$ and $\mathbf{H}_t \in \mathbb{R}_{\geq 0}^{K \times N_t}$. The expectation feature matrix is represented by $\hat{\mathbf{F}}_t = \mathbf{G}_t\mathbf{H}_t$, composed of two non-negative matrices $\mathbf{G}_t \in \mathbb{R}_{\geq 0}^{M \times K}$ and $\mathbf{H}_t \in \mathbb{R}_{\geq 0}^{K \times N_t}$.

Interaction $v_{\mathrm{i}j,t}$ between node i and node j at time t is modeled by a Poisson distribution $\mathbb{P}(\mathbf{v}_{\mathrm{i}j,t} = v_{\mathrm{i}j,t}) = \exp\left(-\hat{v}_{\mathrm{i}j,t}\right)\frac{\hat{v}_{ij,t}^{v_{ij,t}}}{v_{\mathrm{i}j,t}!}$ with rate $\hat{v}_{\mathrm{i}j,t} = \sum_{k=1}^{K} w_{\mathrm{i}k,t}h_{kj,t}$, hence $\mathbb{E}(v_{\mathrm{i}j,t}) = \hat{v}_{\mathrm{i}j,t}$. Value of feature m for node j at time t, $f_{mj,t}$, is modeled by a Poisson distribution $\mathbb{P}(\mathbf{f}_{mj,t} = f_{mj,t}) = \exp\left(-\hat{f}_{mj,t}\right)\frac{\hat{f}_{mj,t}^{f_{mj,t}}}{f_{mj,t!}}$ with rate $\hat{f}_{mj,t} = \sum_{k=1}^{K} g_{mk,t}h_{kj,t}$, hence $\mathbb{E}(f_{mj,t}) = \hat{f}_{mj,t}$.

Inspired by Afsariardchi (2012), we model the transition from $\mathbf{W}_{t-1}$ to $\mathbf{W}_t$ as a Gamma distribution; likewise for the transition from $\mathbf{H}_{t-1}$ to $\mathbf{H}_t$, and $\mathbf{G}_{t-1}$ to $\mathbf{G}_t$. In this case, a half-normal prior distribution over the columns of $\mathbf{W}_t$, rows of $\mathbf{H}_t$, and columns of $\mathbf{G}_t$ is used, with parameters $\boldsymbol{\beta}_t$. This automatic relevance determination process will allow to find the number of communities at each time step. Finally, each $\boldsymbol{\beta}_t$ is modeled as a Gamma distribution.

2.3.2.1. Model for time step 1

Since at the first timestep there is no information for $\mathbf{W}_1$, $\mathbf{H}_1$ and $\mathbf{G}_1$ on previous snapshots, the joint distribution over all variables is expressed by Eq. (2.1) and the posterior by Eq. (2.2).

$$\begin{aligned}\mathbb{P}\left(\mathbf{V}_1, \mathbf{F}_1, \mathbf{W}_1, \mathbf{H}_1, \mathbf{G}_1, \boldsymbol{\beta}_1\right) &= \mathbb{P}\left(\mathbf{V_1}, \mathbf{F_1} | \mathbf{W}_1, \mathbf{H}_1, \mathbf{G}_1, \boldsymbol{\beta}_1\right)\mathbb{P}\left(\mathbf{W}_1, \mathbf{H}_1, \mathbf{G}_1, \boldsymbol{\beta}_1\right) \\ &= \mathbb{P}\left(\mathbf{V}_1 | \mathbf{W}_1, \mathbf{H}_1\right)\mathbb{P}\left(\mathbf{F}_1 | \mathbf{G}_1, \mathbf{H}_1\right)\mathbb{P}\left(\mathbf{W}_1 | \boldsymbol{\beta}_1\right)\mathbb{P}\left(\mathbf{H}_1 | \boldsymbol{\beta}_1\right)\mathbb{P}\left(\mathbf{G}_1 | \boldsymbol{\beta}_1\right)\mathbb{P}\left(\boldsymbol{\beta}_1\right)\end{aligned} \tag{2.1}$$

$$\begin{aligned}&\mathbb{P}\left(\mathbf{W}_1, \mathbf{H}_1, \mathbf{G}_1, \boldsymbol{\beta}_1 | \mathbf{V}_1, \mathbf{F}_1\right) = \\ &\frac{\mathbb{P}\left(\mathbf{V}_1 | \mathbf{W}_1, \mathbf{H}_1\right)\mathbb{P}\left(\mathbf{F}_1 | \mathbf{G}_1, \mathbf{H}_1\right)\mathbb{P}\left(\mathbf{W}_1 | \boldsymbol{\beta}_1\right)\mathbb{P}\left(\mathbf{H}_1 | \boldsymbol{\beta}_1\right)\mathbb{P}\left(\mathbf{G}_1 | \boldsymbol{\beta}_1\right)p\left(\boldsymbol{\beta}_1\right)}{\mathbb{P}\left(\mathbf{V}_1, \mathbf{F}_1\right)}\end{aligned} \tag{2.2}$$

The objective is to maximize the model posterior of Eq.(2.2) given the observations, which can be achieved by minimizing the negative log posterior, shaping the equation into a loss function. Since $\mathbb{P}\left(\mathbf{V}_1, \mathbf{F}_1\right)$ is constant, it does not affect the results of the optimization, so the loss function is defined by Eq. (2.3).

$$\begin{aligned}\mathcal{U}_1 = &-\log\mathbb{P}\left(\mathbf{V}_1 | \mathbf{W}_1, \mathbf{H}_1\right) - \log\mathbb{P}\left(\mathbf{F}_1 | \mathbf{G}_1, \mathbf{H}_1\right) \\ &- \log\mathbb{P}\left(\mathbf{W}_1 | \boldsymbol{\beta}_1\right) - \log\mathbb{P}\left(\mathbf{H}_1 | \boldsymbol{\beta}_1\right) - \log\mathbb{P}\left(\mathbf{G}_1 | \boldsymbol{\beta}_1\right) - \log\mathbb{P}\left(\boldsymbol{\beta}_1\right)\end{aligned} \tag{2.3}$$

After replacing each term and performing all necessary computations, the loss function from Eq. (2.3) transforms to Eq. (2.4), where $a_{k,t}$ and $b_{k,t}$ are fixed hyperparameters, as has been proposed, e.g., in Psorakis et al. (Psorakis et al., 2011). Please refer to Annex A for

further details.

$$\begin{aligned}
&\mathcal{U}_1\left(\mathbf{W}_1, \mathbf{H}_1, \mathbf{G}_1, \boldsymbol{\beta}_1\right) = \\
&\sum_{\mathrm{i}=1}^{N_1}\sum_{j=1}^{N_1}\left(\sum_{k=1}^{K} w_{\mathrm{i}k,1}h_{kj,1} - v_{\mathrm{i}j,1}\log\left(\sum_{k=1}^{K} w_{\mathrm{i}k,1}h_{kj,1}\right)\right) + \\
&\sum_{m=1}^{M}\sum_{j=1}^{N_1}\left(\sum_{k=1}^{K} g_{mk,1}h_{kj,1} - f_{mj,1}\log\left(\sum_{k=1}^{K} g_{mk,1}h_{kj,1}\right)\right) \\
&+\frac{1}{2}\sum_{k=1}^{K}\left(\sum_{\mathrm{i}=1}^{N_1}\beta_{k,1}w_{\mathrm{i}k,1}^2 + \sum_{j=1}^{N_1}\beta_{k,1}h_{kj,1}^2 + \sum_{m=1}^{M} g_{mk,1}^2\beta_{k,1}\right) \\
&+\sum_{k=1}^{K}\left(\beta_{k,1}b_{k,1} - (a_{k,1}-1)\log\beta_{k,1} - \left(N_1 + \frac{M}{2}\right)\log\beta_{k,1}\right) + \kappa
\end{aligned} \tag{2.4}$$

2.3.2.2. Model for time step t

Now that the first snapshot has been defined, successive snapshots are characterized. Since the number of nodes can change between consecutive snapshots due to the birth or death of objects, an equivalence between community assignment matrices $\mathbf{W}_{t-1}$ and $\mathbf{W}_t$, and $\mathbf{H}_{t-1}$ and $\mathbf{H}_t$ must be established. To achieve this, we delete the rows from $\mathbf{W}_{t-1}$ that represent the nodes that disappeared between snapshots $t-1$ and t, resulting in $\mathbf{W}_{t-1}^{'}$. Then, a row is added for each new node that appears at snapshot t, obtaining $\mathbf{W}_{t-1}^{''}$. If a node reappears in the network, the row is added in the appropriate place. A similar approach is used for the columns of $\mathbf{H}_{t-1}$.

Joint distribution over all variables for snapshot t can be expressed by Eq. (2.5) and the posterior by Eq. (2.6), where μ is a parameter of the distribution used to model the transition between successive snapshots.

$$\begin{aligned}
&\mathbb{P}\left(\mathbf{V}_t, \mathbf{F}_t, \mathbf{W}_t, \mathbf{W}_{t-1}^{''}, \mathbf{H}_t, \mathbf{H}_{t-1}^{''}, \mathbf{G}_t, \mathbf{G}_{t-1}, \mu, \boldsymbol{\beta}_t\right) \\
&= \mathbb{P}\left(\mathbf{V_t}, \mathbf{F_t}|\mathbf{W}_t, \mathbf{H}_t, \mathbf{G}_t, \boldsymbol{\beta}_t\right)\mathbb{P}\left(\mathbf{W}_t, \mathbf{W}_{t-1}^{''}, \mathbf{H}_t, \mathbf{H}_{t-1}^{''}, \mathbf{G}_t, \mathbf{G}_{t-1}, \mu, \boldsymbol{\beta}_t\right) \\
&= \mathbb{P}\left(\mathbf{V}_t|\mathbf{W}_t, \mathbf{H}_t\right)\mathbb{P}\left(\mathbf{F}_t|\mathbf{G}_t, \mathbf{H}_t\right) \\
&\mathbb{P}\left(\mathbf{W}_t|\mathbf{W}_{t-1}^{''}, \mu, \boldsymbol{\beta}_t\right)\mathbb{P}\left(\mathbf{H}_t|\mathbf{H}_{t-1}^{''}, \mu, \boldsymbol{\beta}_t\right)\mathbb{P}\left(\mathbf{G}_t|\mathbf{G}_{t-1}, \mu, \boldsymbol{\beta}_t\right)\mathbb{P}\left(\boldsymbol{\beta}_t\right)
\end{aligned} \tag{2.5}$$

$$\begin{aligned}
&\mathbb{P}\left(\mathbf{W}_t, \mathbf{H}_t, \mathbf{G}_t, \boldsymbol{\beta}_t|\mathbf{V}_t, \mathbf{F}_t, \mathbf{W}_{t-1}^{''}, \mathbf{H}_{t-1}^{''}, \mathbf{G}_{t-1}, \mu\right) \\
&= \frac{\mathbb{P}\left(\mathbf{V}_t, \mathbf{F}_t, \mathbf{W}_t, \mathbf{W}_{t-1}^{''}, \mathbf{H}_t, \mathbf{H}_{t-1}^{''}, \mathbf{G}_t, \mathbf{G}_{t-1}, \mu, \boldsymbol{\beta}_t\right)}{\mathbb{P}\left(\mathbf{V}_t, \mathbf{F}_t, \mathbf{W}_{t-1}^{''}, \mathbf{H}_{t-1}^{''}, \mathbf{G}_{t-1}, \mu\right)}
\end{aligned} \tag{2.6}$$

The objective is to maximize the model posterior defined by Eq. (2.6), which can be achieved by minimizing the loss function defined by Eq. (2.7).

$$\begin{aligned}
\mathcal{U}_t = &-\log\mathbb{P}\left(\mathbf{V}_t|\mathbf{W}_t, \mathbf{H}_t\right) - \log\mathbb{P}\left(\mathbf{F}_t|\mathbf{G}_t, \mathbf{H}_t\right) - \log\mathbb{P}\left(\mathbf{W}_t|\mathbf{W}_{t-1}^{''}, \mu, \boldsymbol{\beta}_t\right) \\
&- \log\mathbb{P}\left(\mathbf{H}_t|\mathbf{H}_{t-1}^{''}, \mu, \boldsymbol{\beta}_t\right) - \log\mathbb{P}\left(\mathbf{G}_t|\mathbf{G}_{t-1}, \mu, \boldsymbol{\beta}_t\right) - \log\mathbb{P}\left(\boldsymbol{\beta}_t\right)
\end{aligned} \tag{2.7}$$

Terms one, two and six from Eq. (2.7) are defined in Annex A as Eqs. (A.4), (A.5) and (A.10), respectively.

Following Afsariardchi (2012), the transition between $\mathbf{W}''_{t-1}$ to $\mathbf{W}_t$ is modeled by a Gamma distribution (the same for the transition between $\mathbf{H}''_{t-1}$ to $\mathbf{H}_t$ and $\mathbf{G}_{t-1}$ to $\mathbf{G}_t$), where $p\left(w_{\mathrm{i}k,t}\right) \sim Gamma(\gamma_{\mathrm{i}k,t}, \mu)$, with $\gamma_{\mathrm{i}k,t} = \mu w''_{\mathrm{i}k,t-1} + 1$ as the shape parameter and μ as the rate parameter. Furthermore, half normal priors over the columns of $\mathbf{W}_t$ and $\mathbf{G}_t$, and the rows of $\mathbf{H}_t$, with scale parameters $\boldsymbol{\beta}_t \in \mathbb{R}^K$, are used. Then, $w_{\mathrm{i}k,t} \sim \mathcal{HN}(0, \sigma^2)$, where $\sigma^2 = \beta_{k,t}^{-1}$. Finally, the third term of Eq. (2.7) is determined using Eqs. (B.1) and (B.2) from Annex B. Analogously, the fourth term of Eq. (2.7) is obtained by equation (B.3) and the fifth term is obtained by equation (B.4).

Aggregating all results, the loss function from Eq. (2.7) can be expressed as Eq. (2.8), where $t = 2, \ldots, T$.

$$
\begin{aligned}
\mathcal{U}_t = & \sum_{\mathrm{i}=1}^{N_t} \sum_{j=1}^{N_t} \left(\sum_{k=1}^{K} w_{\mathrm{i}k,t} h_{kj,t} - v_{\mathrm{i}j,t} \log \left(\sum_{k=1}^{K} w_{\mathrm{i}k,t} h_{kj,t} \right) \right) \\
& + \sum_{m=1}^{M} \sum_{j=1}^{N_t} \left(\sum_{k=1}^{K} g_{mk,t} h_{kj,t} - f_{mj,t} \log \left(\sum_{k=1}^{K} g_{mk,t} h_{kj,t} \right) \right) \\
& + \sum_{\mathrm{i}=1}^{N_t} \sum_{k=1}^{K} \mu \left(-w''_{\mathrm{i}k,t-1} \log w_{\mathrm{i}k,t} + w_{\mathrm{i}k,t} \right) + \sum_{\mathrm{i}=1}^{N_t} \sum_{k=1}^{K} \frac{w_{\mathrm{i}k,t}^2 \beta_{k,t}}{2} - \frac{N_t}{2} \sum_{k=1}^{K} \log \beta_{k,t} \\
& + \sum_{j=1}^{N_t} \sum_{k=1}^{K} \mu \left(-h''_{kj,t-1} \log h_{kj,t} + h_{kj,t} \right) + \sum_{j=1}^{N_t} \sum_{k=1}^{K} \frac{h_{kj,t}^2 \beta_{k,t}}{2} - \frac{N_t}{2} \sum_{k=1}^{K} \log \beta_{k,t} \\
& + \sum_{\mathrm{i}=1}^{M} \sum_{k=1}^{K} \mu \left(-g_{mk,t-1} \log g_{mk,t} + g_{mk,t} \right) + \sum_{\mathrm{i}=1}^{M} \sum_{k=1}^{K} \frac{g_{mk,t}^2 \beta_{k,t}}{2} - \frac{M}{2} \sum_{k=1}^{K} \log \beta_{k,t} \\
& + \sum_{k=1}^{K} \left(\beta_{k,t} b_{k,t} - (a_{k,t} - 1) \log \beta_{k,t} \right) + \kappa
\end{aligned}
\tag{2.8}
$$

2.3.3. Iterative solution algorithm

The loss function at each time step has to be minimized. In Tan & Févotte (2009), an iterative procedure for minimizing a loss function is proposed using the gradient of the loss function with respect to the latent factors. This is based on a coordinate descent iterative algorithm for NMF with multiplicative update rules designed by Lee & Seung (1999, 2001). Inspired by this approach we obtain the values of $\mathbf{H}_t$, $\mathbf{W}_t$ and $\mathbf{G}_t$, and scale hyperparameters $\boldsymbol{\beta}_t$.

2.3.3.1. Equations for time step 1

The scalar form of the partial derivative of the loss function from Eq. (2.4) with respect to $\mathbf{H}_1$, $\mathbf{W}_1$, $\mathbf{G}_1$, and $\beta_{k,1}$, is shown in Eqs. (C.1) to (C.4), respectively, from Annex C. The gradient of the loss function with respect to the latent factors in a matrix form is shown in Eqs. (C.5) to (C.7).

The multiplicative coordinate descent algorithm update process for each matrix $\mathbf{H}_1$, $\mathbf{W}_1$ and $\mathbf{G}_1$ is based on multiplying its current value with the ratio of negative to positive part of the gradient of (2.4) with respect to the corresponding matrix (Lee & Seung, 1999, 2001). For $\beta_{k,1}$, the value is chosen with traditional optimization approach, such that the partial derivative with respect to $\beta_{k,1}$ is zero.

Specifically, the update process consists on iterate through equations (2.9), (2.10), (2.11) and (2.12), until a maximum number of iterations or a convergence measure obtained by the frobenius norm of the differences between $\mathbf{V}_1$ and the product of $\mathbf{W}_1$ and $\mathbf{H}_1$, has been reached.

$$\mathbf{H}_{\mathbf{1}K\times N_1} \leftarrow \left(\frac{\mathbf{H}_{\mathbf{1}K\times N_1}}{\left(\mathbf{W}_{\mathbf{1}N_1\times K}\right)^T \mathbf{1}_{N_1\times N_1} + \left(\mathbf{G}_{\mathbf{1}M\times K}\right)^T \mathbf{1}_{M\times N_1} + \mathbf{B}_{\mathbf{1}K\times K}\mathbf{H}_{\mathbf{1}K\times N_1}}\right) \circ \left[\left(\mathbf{W}_{\mathbf{1}N_1\times K}\right)^T \frac{\mathbf{V}_{\mathbf{1}N_1\times N_1}}{\mathbf{W}_{\mathbf{1}N_1\times K}\mathbf{H}_{\mathbf{1}K\times N_1}} + \left(\mathbf{G}_{\mathbf{1}M\times K}\right)^T \frac{\mathbf{F}_{\mathbf{1}M\times N_1}}{\mathbf{G}_{\mathbf{1}M\times K}\mathbf{H}_{\mathbf{1}K\times N_1}}\right] \tag{2.9}$$

$$\mathbf{W}_{\mathbf{1}N_1\times K} \leftarrow \left(\frac{\mathbf{W}_{\mathbf{1}N_1\times K}}{\mathbf{1}_{N_1\times N_1}\left(\mathbf{H}_{\mathbf{1}K\times N_1}\right)^T + \mathbf{W}_{\mathbf{1}N_1\times K}\mathbf{B}_{K\times K}}\right) \circ \left[\frac{\mathbf{V}_{\mathbf{1}N_1\times N_1}}{\mathbf{W}_{\mathbf{1}N_1\times K}\mathbf{H}_{\mathbf{1}K\times N_1}}\left(\mathbf{H}_{\mathbf{1}K\times N_1}\right)^T\right] \tag{2.10}$$

$$\mathbf{G}_{\mathbf{1}M\times K} \leftarrow \left(\frac{\mathbf{G}_{\mathbf{1}M\times K}}{\mathbf{1}_{M\times N_1}\left(\mathbf{H}_{\mathbf{1}K\times N_1}\right)^T + \mathbf{G}_{\mathbf{1}M\times K}\mathbf{B}_{\mathbf{1}K\times K}}\right) \circ \left[\frac{\mathbf{F}_{\mathbf{1}M\times N_1}}{\mathbf{G}_{\mathbf{1}M\times K}\mathbf{H}_{\mathbf{1}K\times N_1}}\left(\mathbf{H}_{\mathbf{1}K\times N_1}\right)^T\right] \tag{2.11}$$

$$\beta_{k,1} \leftarrow \frac{N_1 + M/2 + a_{k,1} - 1}{\frac{1}{2}\left(\sum_{\mathrm{i}=1}^{N_1} w_{\mathrm{i}k,1}^2 + \sum_{j=1}^{N_1} h_{kj,1}^2 + \sum_{m=1}^{M} g_{mk,1}^2\right) + b_{k,1}} \tag{2.12}$$

2.3.3.2. Equations for time step t

Similarly as for time step 1, the scalar form of the partial derivative of the loss function from Eq. (2.8) with respect to $\mathbf{H}_t$, $\mathbf{W}_t$, $\mathbf{G}_t$, and $\beta_{k,t}$, is shown in Eqs. (D.1) to (D.4), respectively, from Annex D. The gradient of the loss function with respect to the latent factors in a matrix form is shown in Eqs. (D.5) to (D.7).

The parameter μ in these equations controls the importance of past information about communities. If $\mu = 0$, no past information about communities is used. If $\mu \to \infty$, past information has the utmost importance. For a better interpretation of the parameter μ, we can replace $\mu = \frac{1-\alpha}{\alpha}$, where $\alpha \in [0, 1]$. Then, as α tends to 0, only past information about communities is used, and if $\alpha = 1$, only the current adjacency matrix is used to find the communities (Afsariardchi, 2012).

Results from applying the multiplicative coordinate descent algorithm are shown in Eqs. (E.1) to (E.3), from Annex E.

These equations can be rewritten, so the update process consists on iterate through Eqs. (2.13) to (2.16), until a maximum number of iterations or a convergence measure obtained by the frobenius norm of the differences between $\mathbf{V}_t$ and the product of $\mathbf{W}_t$ and $\mathbf{H}_t$, has

been reached.

$$\mathbf{H}_{\mathbf{t}K\times N_t} \leftarrow \frac{\alpha\mathbf{H}_{\mathbf{t}K\times N_t} \circ \left(\left(\mathbf{W}_{\mathbf{t}N_t\times K}\right)^T \frac{\mathbf{V}_{\mathbf{t}N_t\times N_t}}{\mathbf{W}_{\mathbf{t}N_t\times K}\mathbf{H}_{\mathbf{t}K\times N_t}} + \left(\mathbf{G}_{\mathbf{t}M\times K}\right)^T \frac{\mathbf{F}_{\mathbf{t}M\times N_t}}{\mathbf{G}_{\mathbf{t}M\times K}\mathbf{H}_{\mathbf{t}K\times N_t}} \right)}{\alpha\left(\mathbf{W}_{\mathbf{t}N_t\times K}\right)^T \mathbf{1}_{N_t\times N_t} + \alpha\left(\mathbf{G}_{\mathbf{t}M\times K}\right)^T \mathbf{1}_{M\times N_t} + (1-\alpha)\,\mathbf{1}_{K\times N_t} + \alpha\mathbf{B}_{\mathbf{t}K\times K}\mathbf{H}_{\mathbf{t}K\times N_t}} + \frac{(1-\alpha)\,\mathbf{H}''_{\mathbf{t-1}K\times N_t}}{\alpha\left(\mathbf{W}_{\mathbf{t}N_t\times K}\right)^T \mathbf{1}_{N_t\times N_t} + \alpha\left(\mathbf{G}_{\mathbf{t}M\times K}\right)^T \mathbf{1}_{M\times N_t} + (1-\alpha)\,\mathbf{1}_{K\times N_t} + \alpha\mathbf{B}_{\mathbf{t}K\times K}\mathbf{H}_{\mathbf{t}K\times N_t}} \tag{2.13}$$

$$\mathbf{W}_{\mathbf{t}N_t\times K} \leftarrow \left(\frac{\alpha\mathbf{W}_{\mathbf{t}N_t\times K} \circ \left(\frac{\mathbf{V}_{\mathbf{t}N_t\times N_t}}{\mathbf{W}_{\mathbf{t}N_t\times K}\mathbf{H}_{\mathbf{t}K\times N_t}} \left(\mathbf{H}_{\mathbf{t}K\times N_t}\right)^T \right) + (1-\alpha)\,\mathbf{W}''_{\mathbf{t-1}N_t\times K}}{\alpha\mathbf{1}_{N_t\times N_t}\left(\mathbf{H}_{\mathbf{t}K\times N_t}\right)^T + (1-\alpha)\,\mathbf{1}_{N_t\times K} + \alpha\mathbf{W}_{\mathbf{t}N_t\times K}\mathbf{B}_{\mathbf{t}K\times K}} \right) \tag{2.14}$$

$$\mathbf{G}_{\mathbf{t}M\times K} \leftarrow \left(\frac{\alpha\mathbf{G}_{\mathbf{t}M\times K} \circ \left(\frac{\mathbf{F}_{\mathbf{t}M\times N_t}}{\mathbf{G}_{\mathbf{t}M\times K}\mathbf{H}_{\mathbf{t}K\times N_t}} \left(\mathbf{H}_{\mathbf{t}K\times N_t}\right)^T \right) + (1-\alpha)\,\mathbf{G}_{\mathbf{t-1}M\times K}}{\alpha\mathbf{1}_{M\times N_t}\left(\mathbf{H}_{\mathbf{t}K\times N_t}\right)^T + (1-\alpha)\,\mathbf{1}_{M\times K} + \alpha\mathbf{G}_{\mathbf{t}M\times K}\mathbf{B}_{\mathbf{t}K\times K}} \right) \tag{2.15}$$

$$\beta_{k,t} \leftarrow \frac{N_t + M/2 + a_{k,t} - 1}{\frac{1}{2}\left(\sum_{\mathrm{i}=1}^{N_t} w_{\mathrm{i}k,t}^2 + \sum_{m=1}^{M} g_{mk,t}^2 + \sum_{j=1}^{N_t} h_{kj,t}^2\right) + b_{k,t}} \tag{2.16}$$

2.3.4. Full algorithm of OCoDeDANet

Using the previous equations, we define OCoDeDANet to detect overlapping and disjoint communities for a dynamic network, using information about both the graph and nodes' features. The pseudocode of OCoDeDANet is shown in Algorithm 4.

Following the process from lines 1-13 of Algorithm 4, we obtain $\mathbf{H}_t$ for each timestep. Then, communities are determined as described next. Firstly, the rows of $\mathbf{H}_t$ with zero values on all entries are deleted. The number of rows remaining will represent the number of communities at timestamp t. If the algorithm is being used to detect disjoint communities, node j will belong to community k if $h_{kj,t}$ is the greater value of the vector $h_{*j,t}$. If the algorithm is being used to detect overlapping communities, node j will belong to community k if $h_{kj,t}$ is greater than the threshold δ.

In the next section, different datasets are used to evaluate the performance of OCoDeDANet in comparison with other state-of-the-art disjoint and overlapping community detection algorithms.

2.4. Experimental results and evaluation

The effectiveness of OCoDeDANet was tested on disjoint and overlapping synthetic attributed networks with ground truth and one real-world attributed social network without ground truth[1].

[1]The datasets used in this paper are available at `https://data.mendeley.com/datasets/2s75kgnzd7`

Algorithm 4: Full algorithm of OCoDeDANet.

Input: $\mathbf{V}_1, \ldots, \mathbf{V}_T$, $\mathbf{F}_1, \ldots, \mathbf{F}_T$, K, a, b, α, δ, $MinIter$, $MaxIter$, $tolerance$
Output: Clustering labels of each node at each time step, number of communities $K_1^*, \ldots, K_T^*$

Initialize $\mathbf{H}_1$, $\mathbf{W}_1$, $\mathbf{G}_1$ randomly from a uniform distribution $[0, 1]$.
Initialize $\mathbf{B}_1$ as an identity matrix.
for *iter=1 to MaxIter* **do**
    Determine $\mathbf{H}_1$, $\mathbf{W}_1$ and $\mathbf{G}_1$, by Eqs. (2.9) to (2.11).
    Determine $\beta_{k,1}$ $\forall k = 1, \ldots, K$ using Eq. (2.12).
for *t=2 to T* **do**
    Delete the rows (columns) from $\mathbf{W}_{t-1}$ ($\mathbf{H}_{t-1}$) that represent the nodes that disappeared from snapshot transition $t-1$ to t, to obtain $\mathbf{W}'_{t-1}$ ($\mathbf{H}'_{t-1}$).
    Add a row (column) in $\mathbf{W}'_{t-1}$ ($\mathbf{H}'_{t-1}$) for each new node that appears at snapshot t, to obtain $\mathbf{W}''_{t-1}$ ($\mathbf{H}''_{t-1}$).
    Set $iter = 1$.
    repeat
        Determine $\mathbf{H}_t$, $\mathbf{W}_t$ and $\mathbf{G}_t$, by Eqs. (2.13)-(2.15).
        Determine $\beta_{k,t}$ $\forall k = 1, \ldots, K$ using Eq. (2.16).
    until *iter > MaxIter* ***or*** *(iter > MinIter* ***and*** *difference < tolerance)*
for *t=1 to T* **do**
    Delete rows of $\mathbf{H}_t$ with zero values on all entries.
    Assign K_t^* as the number of rows of $\mathbf{H}_t$.
    for $j = 1$ *to* N_t **do**
        For disjoint community detection, assign node j to community k if $h_{kj,t}$ is the greater value of the vector $h_{*j,t}$.
        For overlapping community detection, firstly assign all nodes as in the disjoint case, and then assign node j to community k if $h_{kj,t}$ is greater than the threshold δ.

Four artificial attributed types of networks were generated along with their respective ground-truth information. Three of them disjoint, while one of them overlap. Disjoint networks are similar to the ones described in Márquez & Weber (2023). To demonstrate the performance of our method in the absence of ground truth, we employed one real-world attributed social network.

For the disjoint case, the OCoDeDANet algorithm is compared to the approach of Bello et al. (2016) (named DALouvain in our work), which includes attributes for community detection on dynamic networks, and it is based on modularity optimization[2]. We also compare OCoDeDANet with a version of our model that does not include attributes, and for displaying purposes we called it DBNMF, for Dynamic Bayesian Non-negative Matrix Factorization.

For the overlapping case, the OCoDeDANet algorithm is compared to the static algorithm DEMON proposed by Coscia et al. (2012), and the dynamic algorithm Tiles from Rossetti et al. (2017), both based on label propagation. These algorithms were used as they appear in Rossetti et al. (2019). We also compare OCoDeDANet with DBNMF.

DALouvain's parameters α and β were set as 0.5 as described in Bello et al. (2016). The ε parameter in DEMON's algorithm was set as 0.5. The implementation from Tiles in Rossetti et al. (2019) did not require any parameters.

The structure of our findings is as follows. An overview of the evaluation metrics employed is presented in Section 2.4.1. Subsequently, in Section 2.4.2, we describe various disjoint attributed networks and analyze the performance of different algorithms. Afterward, in Sec-

[2]DALouvain implementation is available at `https://bitbucket.org/harenbergsd/dynamic-attributed-louvain/src/master/`

tion 2.4.3, we describe various overlapping attributed networks and assess the performance of different algorithms. Next, in Section 2.4.4 we show the performance of our model on a real-world network. Finally, Section 2.4.5 briefly summarizes the obtained results.

2.4.1. Performance metrics in community detection

The performance evaluation of community detection algorithms involves various metrics, which depend on the algorithm's characteristics and the network's nature. When ground-truth information about the community structure is available, one commonly used metric is normalized mutual information (NMI). In cases where ground-truth data is absent, other metrics like modularity and density are employed to discover community structure (Chakraborty et al., 2017; Chunaev, 2020). Additionally, when the network has categorical attributed data, entropy can be used to measure the cohesiveness of the network (Chunaev, 2020).

In the case of overlapping community detection with ground-truth information, an overlapping version of the normalized mutual information (ONMI) can be used (Lancichinetti et al., 2009; Rossetti et al., 2019).

Normalized mutual information, is a metric that quantifies the quality of clustering by comparing the detected communities with the true partition. Let $\boldsymbol{N}$ be a confusion matrix, where rows represent class labels from the ground truth, and columns correspond to communities identified by a community detection algorithm. Each element N_{ij} in this matrix indicates the number of nodes with class label i found in community j. The NMI is defined as Eq. (2.17) (Danon et al., 2005). A NMI value of 1 indicates that partitions are identical and a value of 0 indicates that partitions are independent.

$$NMI(\mathcal{A},\mathcal{C}) = \frac{-2\sum_{i=1}^{n_\mathcal{A}}\sum_{j=1}^{n_\mathcal{C}} N_{ij}\log\left(N_{ij}N/\left(N_{i.}N_{.j}\right)\right)}{\sum_{i=1}^{n_\mathcal{A}} N_{i.}\log\left(N_{i.}/N\right)+\sum_{j=1}^{n_\mathcal{C}} N_{.j}\log\left(N_{.j}/N\right)} \tag{2.17}$$

Where:

- A represents the ground-truth partition.
- C represents the detected partition.
- $n_\mathcal{A}$ is the number of ground-truth communities.
- $n_\mathcal{C}$ is the number of found communities.
- $N_{i.}$ is the sum over row i, indicating the number of nodes with class label i.
- $N_{.j}$ is the sum over column j, representing the number of nodes in community j.

Overlapping normalized mutual information is an extension that quantifies the quality of clustering by comparing the detected communities with the true cover, i.e., when overlapping clusters exists. In this work we will use the implementation of Rossetti et al. (2019) which uses the ONMI version of Lancichinetti et al. (2009).

The *density* of a partition $\mathcal{C}$ in a graph is defined as shown in Eq. (2.18) (Dang & Viennet, 2012). The density metric quantifies the proportion of community internal links relative to the total number of links in the graph. Higher density values indicate a better partitioning

of the graph into communities.

$$Density(\mathcal{C}) = \sum_{k=1}^{K} \frac{m_k}{m} \tag{2.18}$$

Where:

- m_k represents the number of internal edges within community k.
- m denotes the total number of edges in the entire graph.
- K is the total number of communities.

The *entropy* of a partition $\mathcal{C}$ measures the similarity of nodes within each community based on the values of a categorical attribute m. The formula for entropy within community k is given by Eq. (2.19) (Dang & Viennet, 2012).

$$entropy(X_m, \mathcal{C}_k) = -\sum_{n=1}^{n_m} p_{mkn} \log(p_{mkn}) \tag{2.19}$$

Where:

- X_m represents a vector of values for attribute m.
- $\mathcal{C}_k$ comprises the nodes belonging to community k.
- n_m is the number of distinct values for attribute m.
- p_{mkn} is the percentage of nodes in community k with a value X_{mn} for attribute m.

The overall entropy of the partition $\mathcal{C}$ is given by Eq. (2.20) (Dang & Viennet, 2012). Lower entropy values indicate greater homogeneity of nodes within each community based on attribute values.

$$Entropy(\mathcal{C}) = \sum_{m=1}^{M} \frac{\alpha_m}{\sum_{p=1}^{M} \alpha_m} \sum_{k=1}^{K} \frac{N_k}{N} \cdot entropy(X_m, \mathcal{C}_k) \tag{2.20}$$

Where:

- N_k represents the number of nodes in community k.
- N is the total number of nodes in the graph.
- α_m is a weight associated with attribute m (if applicable).

2.4.2. Experiments on disjoint synthetic networks

Ten instances were generated using different random seeds for each synthetic network and parameter setting.

OCoDeDANet's parameters for the disjoint version are $\alpha = 0{,}5$ and $K = 50$. The other parameters are set as $a = 5$, $b = 3$, $MinIter = 200$, $MaxIter = 1000$. The parameters set for DBNMF's algorithm were the same. Both algorithms were executed with ten different random initializations of the non-negative matrices.

Attributes were added as follows. For a network of c communities, each one was assumed to have a strong correlation with s binary attributes and a weak correlation with $s \cdot (c-1)$ binary attributes. We use $s = 1$ and $s = 3$ for our tests. The probabilities of having a strong correlation, p_{intra}, were varied from 0.5 to 1, while the probabilities of having a weak correlation, p_{inter}, were set with 0.05 or 0.1 (Chang et al., 2019).

For all synthetic networks, considering $p_{inter} = 0{,}05$, the following configurations will be used: Case 1 is set as $s = 1$ and $p_{intra} = 0{,}7$, Case 2 is set as $s = 3$ and $p_{intra} = 0{,}7$, Case 3 is set as $s = 1$ and $p_{intra} = 1$, and Case 4 is set as $s = 3$ and $p_{intra} = 1$.

Next, we describe the three disjoint synthetic networks created and assess the performance of the different algorithms.

2.4.2.1. Synthetic network 1

We utilize a synthetic benchmark named DANCer (Largeron et al., 2017) to produce attributed graphs. To create each network, the first graph is built by incorporating properties like preferential attachment, small world, or homophily. This graph is then altered using micro and macro operations. The micro operations involve adding or removing nodes and edges. The macro operations involve splitting communities, merging communities, and transferring members to another existing community or a new one (Largeron et al., 2017).

Networks

Table 2.2 shows the parameters for the networks generated with this benchmark, along with a brief description for each one. Over time networks evolve, reaching a maximum value of 1868 nodes, 10 communities and 11905 edges among the built networks. In alignment with the specifics parameters from Table 2.3, a set of twelve datasets were built (Márquez & Weber, 2023).

Results

The results for 6 of the 12 built networks are shown in Fig. 2.2, where the attributes were generated as Case 1. On the available code, DALouvain cannot remove nodes because there is no tracking of the node ID, so the results for DALouvain are obtained in a static manner when $R_{rv} \neq 0$. For OCoDeDANet and DBNMF, the results for this synthetic network include the standard deviation at each time step due to different initialization of non-negative matrices. However, for simplicity, the standard deviation will be omitted for the forthcoming networks.

Over the tested networks, OCoDeDANet performs best in all cases, followed by DBNMF and DALouvain. The difference between OCoDeDANet and DBNMF is similar over time, while DALouvain tends to be closer to DBNMF at the beginning but tends to have a greater performance decay over time. When the probability of merging, splitting and migrate communities increase from 0.3 to 0.8 (Fig. 2.2a to Fig. 2.2b, Fig. 2.2c to Fig. 2.2d, Fig. 2.2e to Fig. 2.2f), the enhanced performance of OCoDeDANet suggests that its ability to capture dynamic behavior increases as changes in these operations increase. A similar behaviour, with greater difference in performance by OCoDeDANet, is shown in Fig. 2.3, when attributes were generated as Case 4.

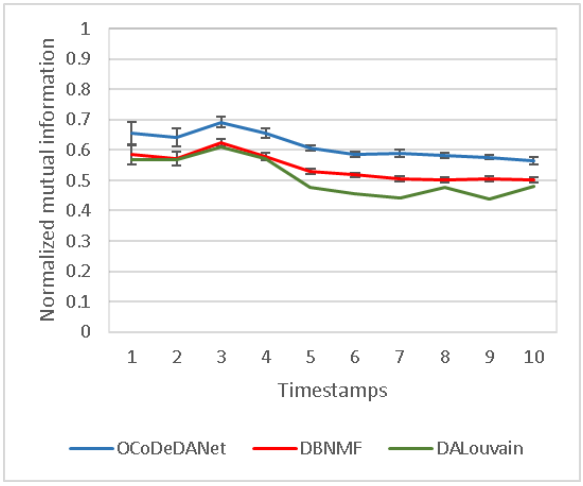
Table 2.2: Parameter values that were used for benchmark DANCer. The description is shown as defined by Largeron et al. (2017).

Parameter	Values	Description
N_{init}	256	Number of nodes at $t = 0$
K_{init}	5	Number of communities at $t = 0$
$NbRep$	5	Maximum number of representatives of each community
P_{rc}	0	A threshold to decide whether a new vertex joins a randomly selected community or not
E_{wth}^{max}	10	Maximum number of edges connecting a new vertex to vertices in its community
E_{btw}^{max}	5	Maximum number of edges connecting a new vertex to vertices in a different community
MTE	1000	Minimum number of total edges
P_{micro}	0.5	A threshold to select if the micro dynamic updates are performed or not
R_{awe}	0.3	Ratio defining the number of within edges inserted
R_{rbe}	0.3	Ratio defining the number of between edges removed
R_{abe}	0.5, 0.9	Ratio defining the number of between edges inserted
R_{rwe}	0.5, 0.9	Ratio defining the number of within edges removed
R_{av}	0.2, 0.3, 0.8	Ratio defining the number of vertices inserted
R_{rv}	0.3, 0, 0.9	Ratio defining the number of vertices removed
P_{merge}	0.3, 0.8	Probability to perform the merge operation
P_{split}	0.3, 0.8	Probability to perform the split operation
$P_{migrate}$	0.3, 0.8	Probability to perform the migrate vertices operation
T	10, 10, 5	Number of graphs generated

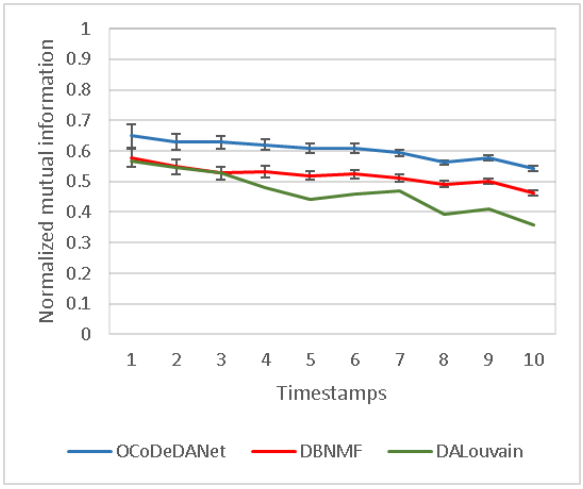
Table 2.3: Specific parameters for networks defined using the benchmark DANCer (Márquez & Weber, 2023).

Dataset	R_{rwe}, R_{abe}	R_{av}	R_{rv}	P_{merge}, P_{split}, $P_{migrate}$	T
1	0.5	0.2	0.3	0.3	10
2	0.5	0.2	0.3	0.8	10
3	0.9	0.2	0.3	0.3	10
4	0.9	0.2	0.3	0.8	10
5	0.5	0.3	0	0.3	10
6	0.5	0.3	0	0.8	10
7	0.9	0.3	0	0.3	10
8	0.9	0.3	0	0.8	10
9	0.5	0.8	0.9	0.3	5
10	0.5	0.8	0.9	0.8	5
11	0.9	0.8	0.9	0.3	5
12	0.9	0.8	0.9	0.8	5

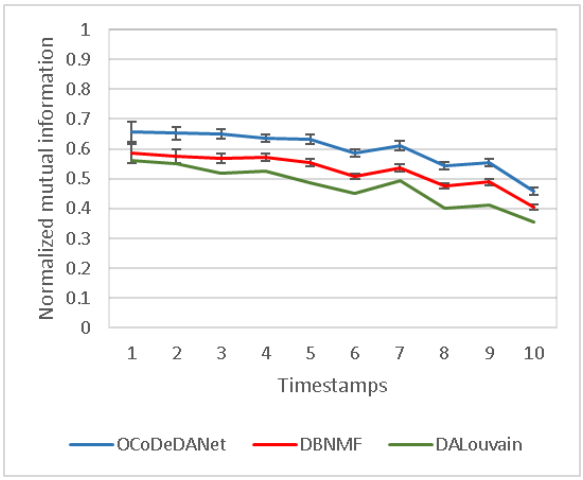
For further analysis of performance, we changed the parameters s, p_{intra} and p_{inter}, and compared the average of NMI for networks 4 and 8, which findings are displayed in Fig. 2.4 and Fig. 2.5, respectively. Results show that as the p_{inter} decreases, i.e., the probability of the attributes being relevant according to the communities is lower, the performance of OCoDeDANet decreases, to the point that in some cases when $p_{intra} = 0{,}5$ the outcome is the same as in DBNMF. According to this tendency, one could argue that when $p_{inter} = 0{,}1$ values of p_{intra} less than 0.5 will make OCoDeDANet worse than DBNMF. Furthermore, we can see that the slope of decrease on performance of OCoDeDANet is greater than in DALouvain.



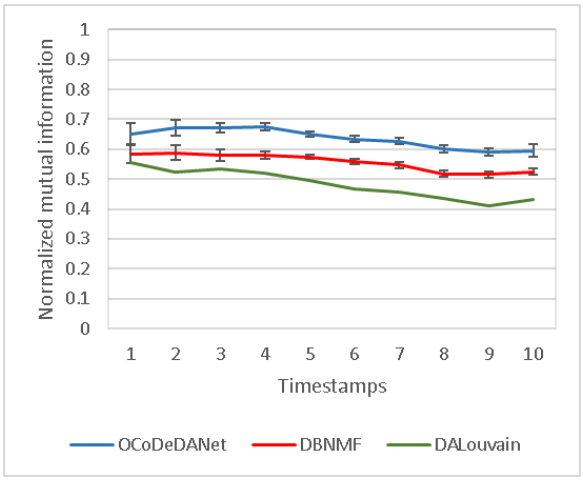
(a) Results for Dataset 3



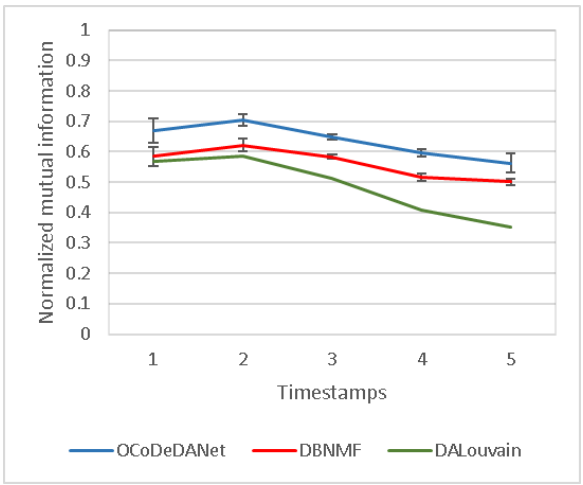
(b) Results for Dataset 4



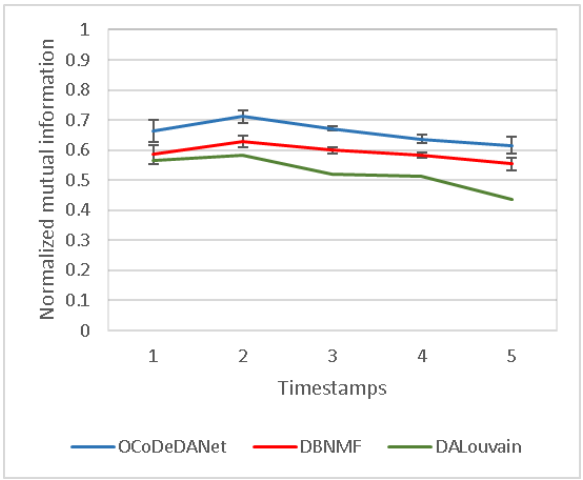
(c) Results for Dataset 7



(d) Results for Dataset 8

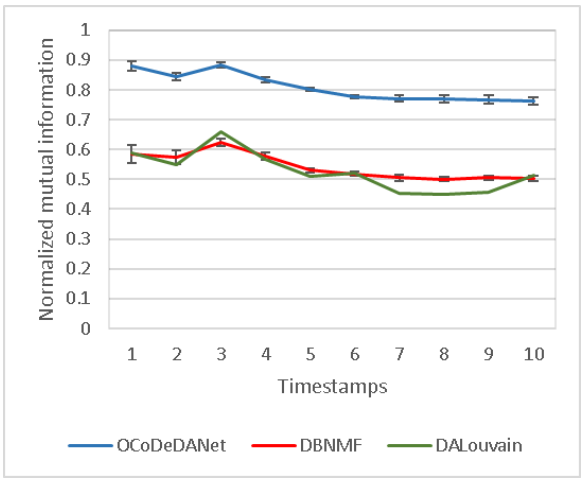


(e) Results for Dataset 11

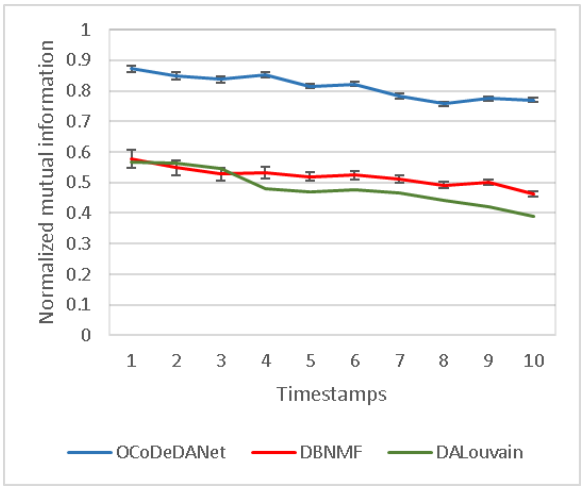


(f) Results for Dataset 12

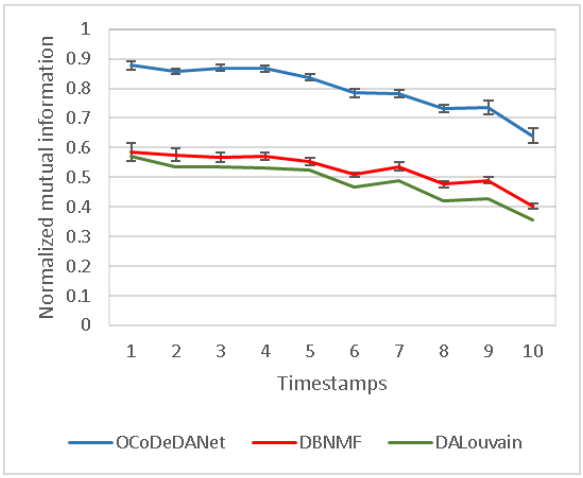
Figure 2.2: Performance measured by NMI for benchmark DANCer when $s = 1$, $p_{intra} = 0{,}7$ and $p_{inter} = 0{,}05$.



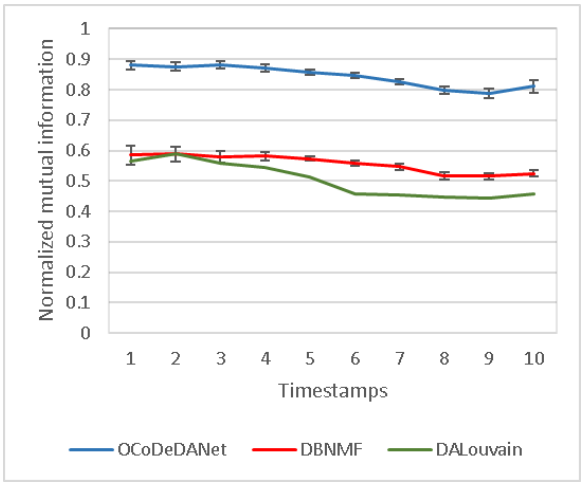
(a) Results for Dataset 3



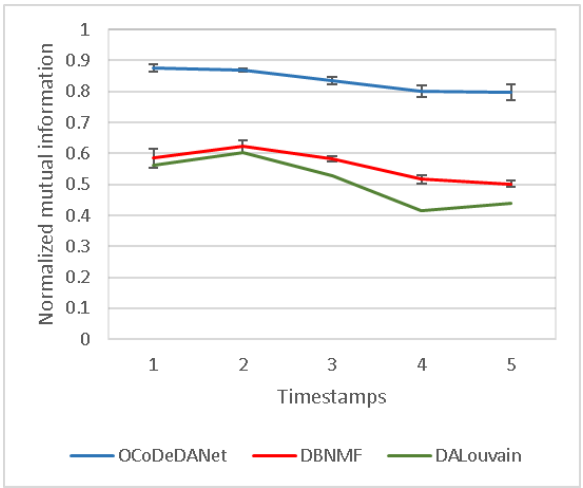
(b) Results for Dataset 4



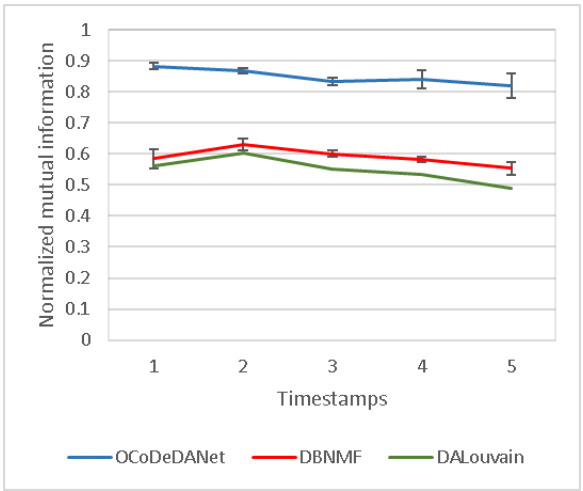
(c) Results for Dataset 7



(d) Results for Dataset 8

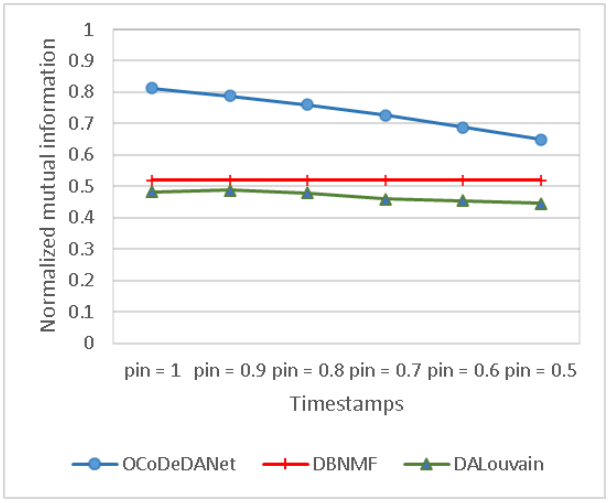


(e) Results for Dataset 11

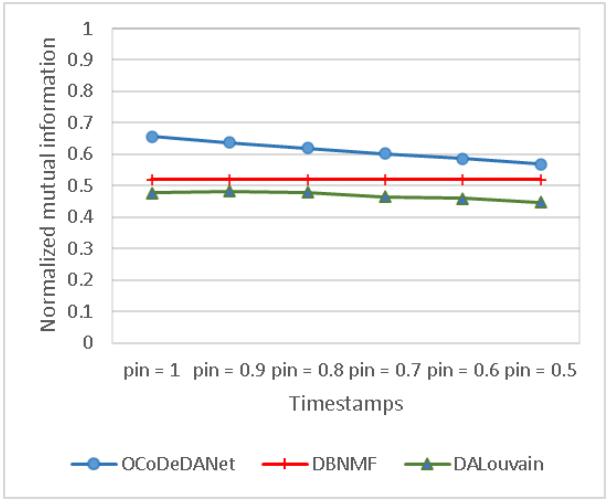


(f) Results for Dataset 12

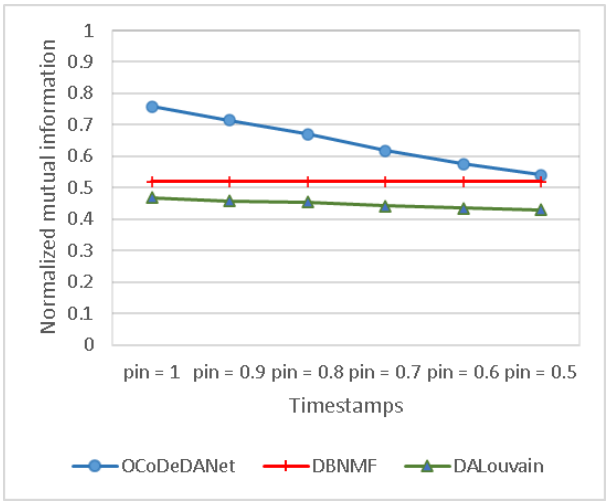
Figure 2.3: Performance measured by NMI for benchmark DANCer when $s = 3$, $p_{intra} = 1$ and $p_{inter} = 0{,}05$.



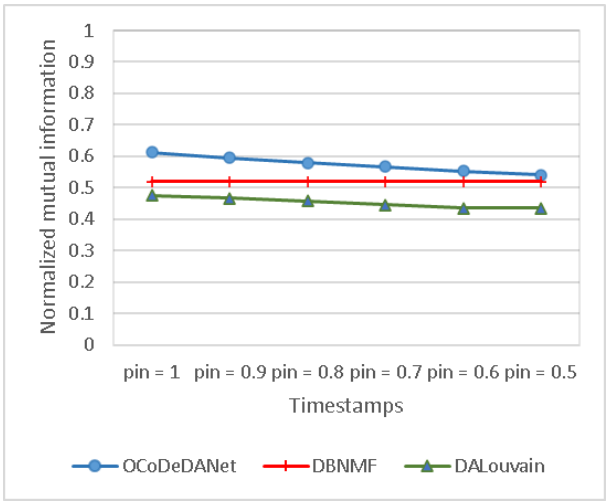
(a) $s = 3$, $p_{inter} = 0{,}05$



(b) $s = 1$, $p_{inter} = 0{,}05$

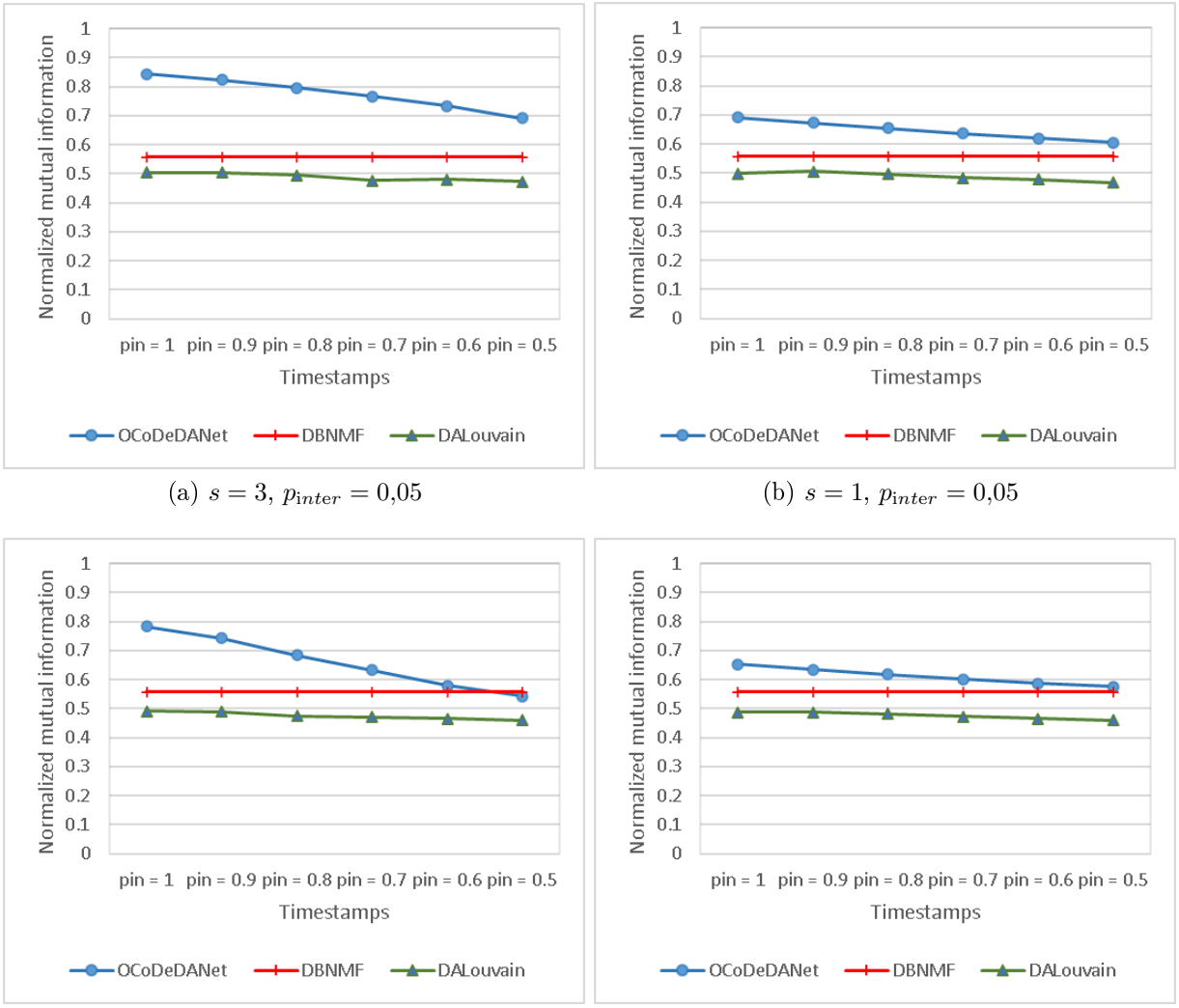


(c) $s = 3$, $p_{inter} = 0{,}1$



(d) $s = 1$, $p_{inter} = 0{,}1$

Figure 2.4: Performance for Dataset 4 of Synthetic network 1 measured by NMI, when p_{intra} is varied between 1 and 0.5.



(a) $s = 3$, $p_{inter} = 0{,}05$

(b) $s = 1$, $p_{inter} = 0{,}05$

(c) $s = 3$, $p_{inter} = 0{,}1$

(d) $s = 1$, $p_{inter} = 0{,}1$

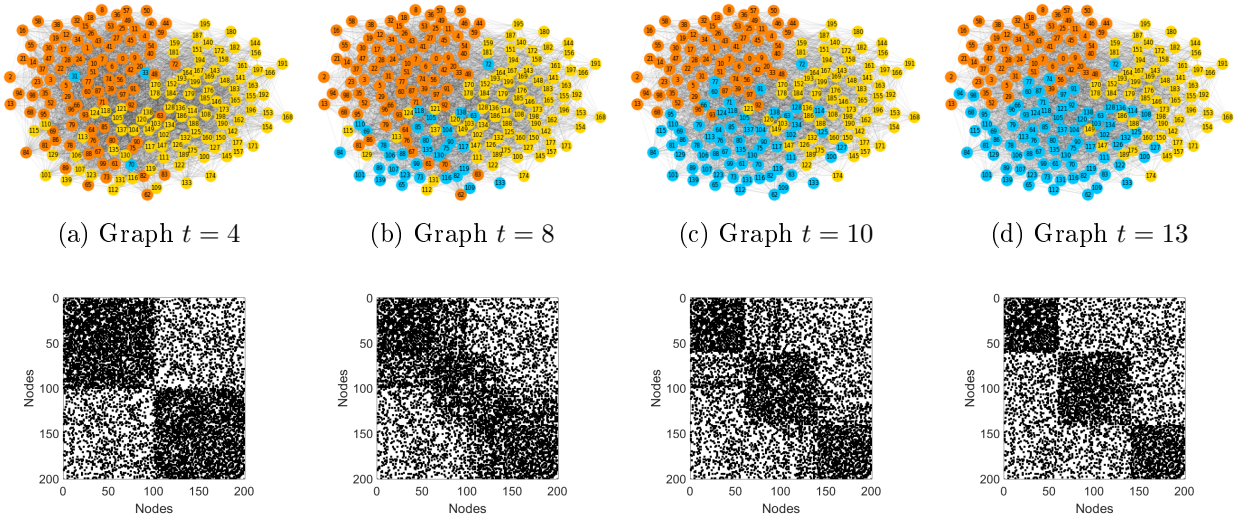
Figure 2.5: Performance for Dataset 8 of Synthetic network 1 measured by NMI, when p_{intra} is varied between 1 and 0.5.

2.4.2.2. Synthetic network 2

Networks

Synthetic network 2 features a network structure with 200 nodes and 20 snapshots. We use this network to show the birth of a community. Synthetic dynamic networks based on Sheikholeslami & Giannakis (2018) and Márquez & Weber (2023) were used. The graphs are built with two communities initially, each with 100 nodes, and a third community appears later. Two independent datasets were built with 40 % and 80 % of nodes migrating to new communities, named Dataset 1 and Dataset 2, respectively. The change in the connections of each node, to allow the new community membership, is made in a specific time selected according to a normal distribution $\mathcal{N}(10, 2)$. A stochastic block model was used to create links between nodes, where nodes within the same community were more likely to be connected (probability of 0.3) than nodes from different communities (probability of 0.1).

Fig. 2.6 shows the graphs and an adjacency matrix representation of selected timestamps for an instance of Dataset 1 and Case 1. The communities found by OCoDeDANet using seed 2 for generation of the data, and seed 1 for initialization of non-negative matrices, are distinguished by colors in the upper part of the figure. In the lower part of the figure, a black dot represents a link between nodes on the horizontal and vertical axes. The network's evolution indicates that certain nodes begin to modify their connections, which support the emergence of a new community.



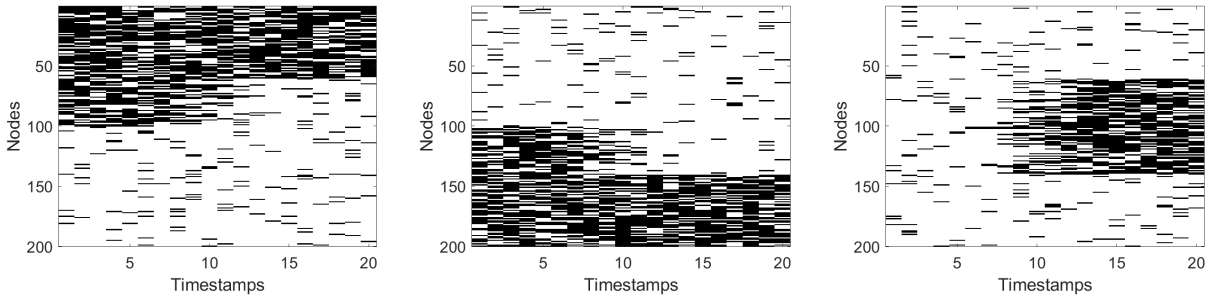
(a) Graph $t = 4$ (b) Graph $t = 8$ (c) Graph $t = 10$ (d) Graph $t = 13$

(e) Adj. $t = 4$ (f) Adj. $t = 8$ (g) Adj. $t = 10$ (h) Adj. $t = 13$

Figure 2.6: An instance of graphs and their corresponding adjacency matrices for Dataset 1, Case 1, of Synthetic network 2.

The attribute values over time are depicted in Fig. 2.7, where a value of 1 is represented by a black color and a value of 0 is represented by white. Attributes 1 and 2 support the identification of two communities at the beginning, but as they change and alongside Attribute 3, a third community can be identified.

Results



(a) Attribute 1 (b) Attribute 2 (c) Attribute 3

Figure 2.7: Evolution of attributes over time, for Dataset 1, Case 1, of Synthetic network 2.

The performance of the models for Dataset 1 of Synthetic network 2 is shown in Fig. 2.8. This figure shows that the performance of OCoDeDANet and DBNMF starts to decay around timestamp 6, and only partially recovers around time step 11. For DALouvain, when $p_{intra} = 1$, there is a considerable decay of performance at timestamp 3 with a maximum recovery around time step 12, followed by another decay; when $p_{intra} = 0{,}7$, the NMI is lower than in other methods since the beginning.

OCoDeDANet shows the best result, closely followed by DBNMF. When there is only one attribute that reinforce the topology, around timestamp 13 both OCoDeDANet and DBNMF behave almost the same way, and from time step 18 DBNMF outperforms OCoDeDANet. At timestamps 1, 2, 11 and 12, despite having the worst average performance, DALouvain outperforms OCoDeDANet and DBNMF when $s = 1$ and $p_{intra} = 1$, and outperforms DBNMF when $s = 3$ and $p_{intra} = 1$.

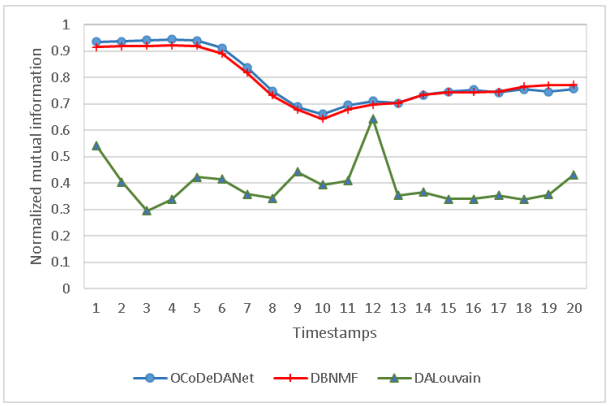
The graph in Fig. 2.9 displays the performance of the models for Dataset 2 of Synthetic network 2. Similar as before, the results indicate that both OCoDeDANet and DBNMF show a decline in performance starting at timestamp 6, but in this case with a higher decrease, reaching NMI values as low as 0.37. Around time step 12 both algorithm start to improve NMI values. DALouvain reaches an NMI average value as low as 0.25.

In this case, beside time steps 1 and 2 for $p_{intra} = 1$, OCoDeDANet shows the best result, followed by DBNMF, both of them showing similar tendencies, and finally DALouvain which exhibits the worst performance.

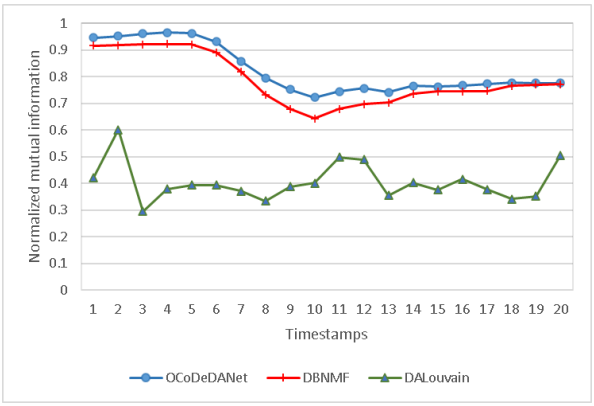
2.4.2.3. Synthetic network 3

Networks

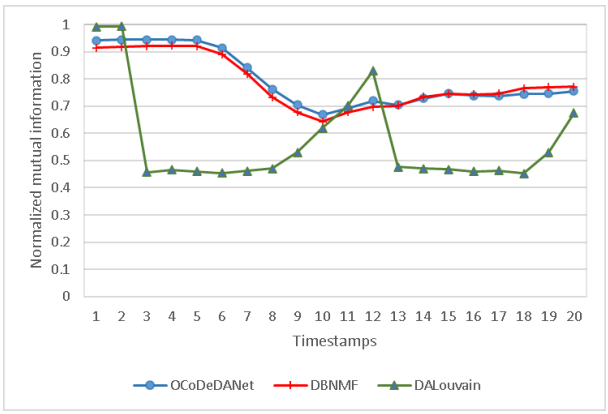
In Synthetic network 3, we conducted a study on community growth and shrinkage, utilizing networks generated with a degree-corrected stochastic block model (Tang et al., 2020). In our experiment, we produced three data sets with ten timestamps, each with three groups of nodes ($n_1 = 80$, $n_2 = 90$, and $n_3 = 100$). We defined the link probability as $z \cdot (\theta_\mathrm{i}) \cdot (\theta_j)$, where $z \in [0, 1]$ represents the original probability of linking nodes i and j in a specific group, and θ_i and θ_j are the degree-corrected parameters. For 95 % of the nodes, θ_i was set to 1, while for the remaining nodes, it was set to 3 (Márquez & Weber, 2023).



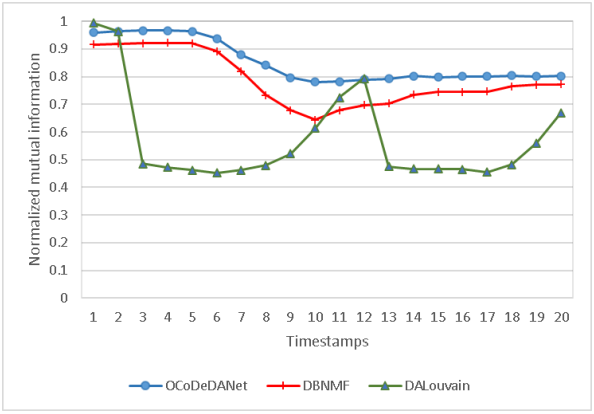
(a) Case 1, $s = 1$, $p_{intra} = 0{,}7$



(b) Case 2, $s = 3$, $p_{intra} = 0{,}7$



(c) Case 3, $s = 1$, $p_{intra} = 1$



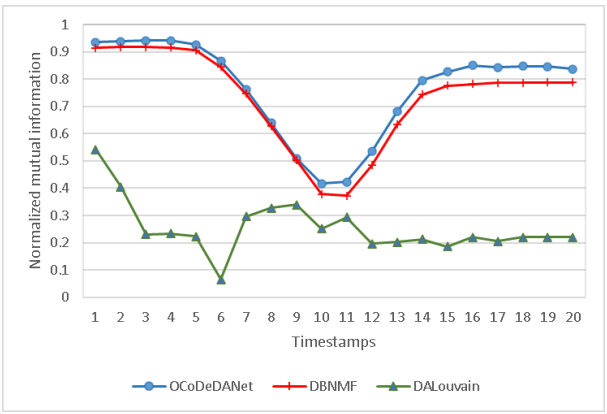
(d) Case 4, $s = 3$, $p_{intra} = 1$

Figure 2.8: Performance for Dataset 1 of Synthetic network 2 (40 % of the nodes migrate to a new community) measured by NMI.

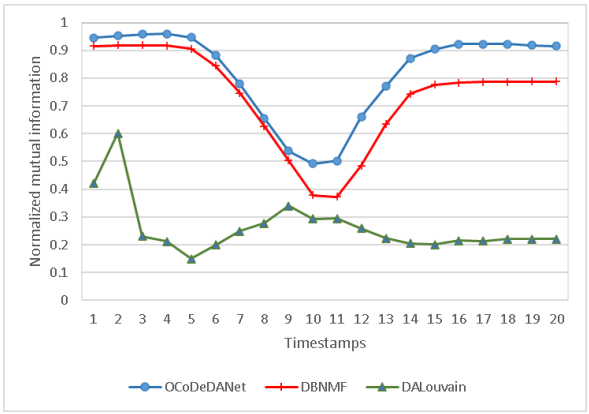
To evaluate strongly assortative structures, weakly assortative structures, and disassortative structures, three types of attributed networks were proposed, each with different original link probabilities. These networks were assessed using the parameter values proposed in Tang et al. (2020) and Márquez & Weber (2023), which are shown in Table 2.4, where $p_1 \in [0{,}21, 0{,}25]$, $p_2 = p_1 - 0{,}01$, $p_3 = p_1 - 0{,}02$, $q \in [0{,}1, 0{,}14]$, and $r \in [0{,}19, 0{,}23]$.

Table 2.4: Original link probabilities in datasets of Synthetic network 3 (Márquez & Weber, 2023)

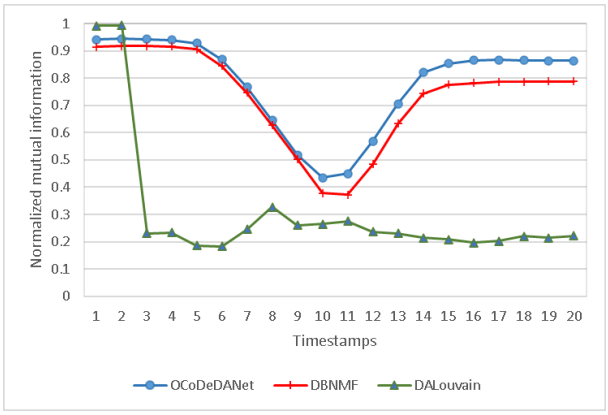
	Within groups link probabilities			Between groups link probabilities		
	1	2	3	1-2	1-3	2-3
Strongly assortative	p_1	p_2	p_3	0.08	0.08	0.08
Weakly assortative	0.3	0.3	0.1	q	0.09	0.09
Disassortative	0.1	0.1	0.1	r	r	r



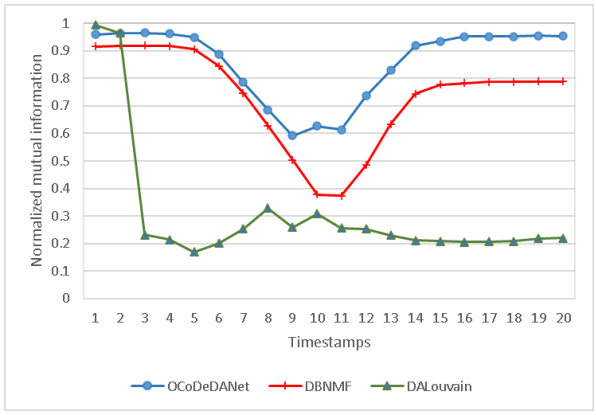
(a) Case 1, $s = 1$, $p_{intra} = 0{,}7$



(b) Case 2, $s = 3$, $p_{intra} = 0{,}7$



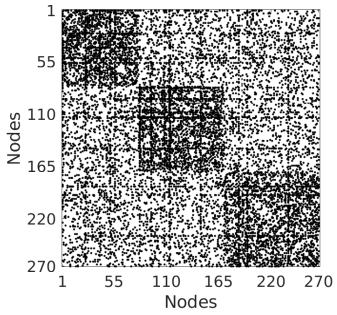
(c) Case 3, $s = 1$, $p_{intra} = 1$



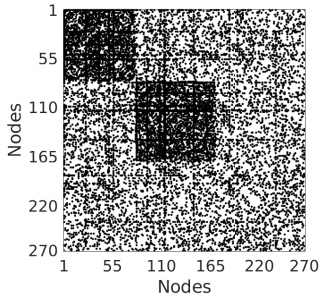
(d) Case 4, $s = 3$, $p_{intra} = 1$

Figure 2.9: Performance for Dataset 2 of Synthetic network 2 (80 % of the nodes migrate to a new community) measured by NMI.

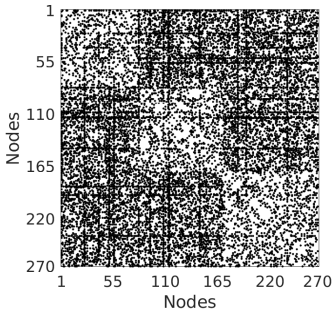
An instance of the adjacency matrices at time step 1 on each type of network, is shown in Fig. 2.10, with parameters $p_1 = 0{,}21$, $q = 0{,}1$ and $r = 0{,}19$ (Márquez & Weber, 2023).



(a) Strongly assortative



(b) Weakly assortative

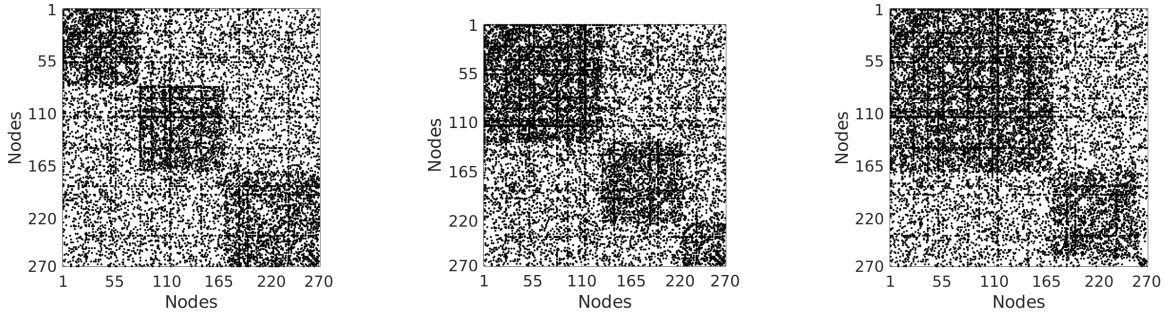


(c) Disassortative

Figure 2.10: An instance of the adjacency matrices for the three types of Synthetic network 3 at $t = 1$ (Márquez & Weber, 2023).

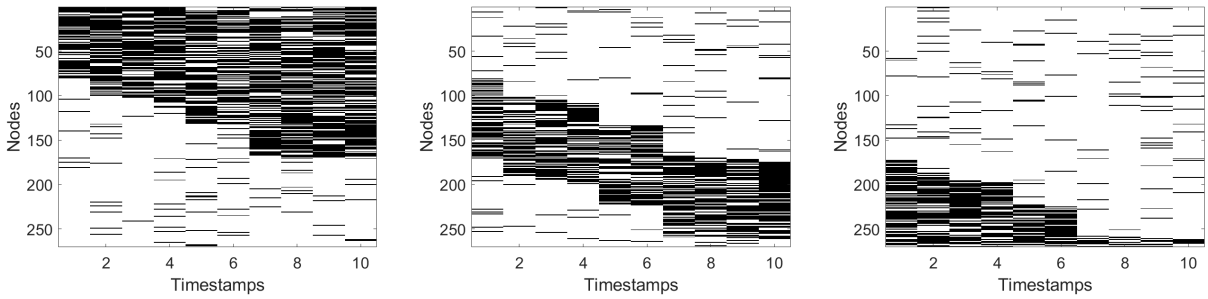
Over the course of 10 time steps, the first community on a strongly assortative network

with $p = 0{,}21$ shows growth, while the second community underwent member changes but maintained its size. Meanwhile, the third community shrank due to changes in both links and attributes. This is illustrated in Fig. 2.11 and Fig. 2.12.



(a) Adj. $t = 1$ (b) Adj. $t = 5$ (c) Adj. $t = 10$

Figure 2.11: An instance of evolution for adjacency matrices, when $s = 3$, $p_{intra} = 0{,}7$, $p_{inter} = 0{,}05$, for the strongly assortative network with $p = 0{,}21$ (Márquez & Weber, 2023).



(a) Attribute 1 (b) Attribute 2 (c) Attribute 3

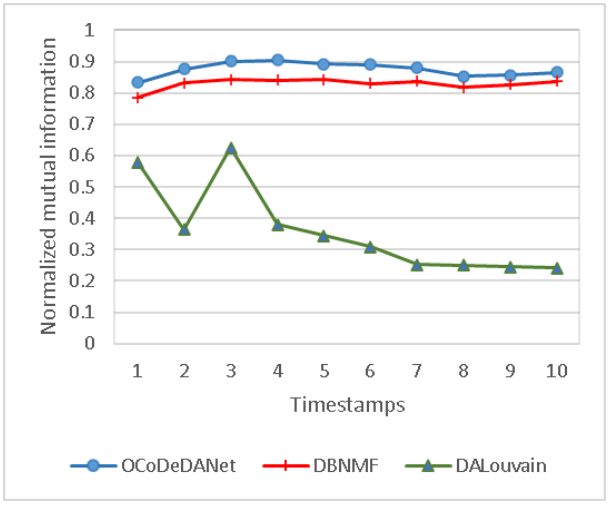
Figure 2.12: An instance of evolution for attributes 1, 2 and 3, when $s = 3$, $p_{intra} = 0{,}7$, $p_{inter} = 0{,}05$, for the strongly assortative network with $p = 0{,}21$.

Results

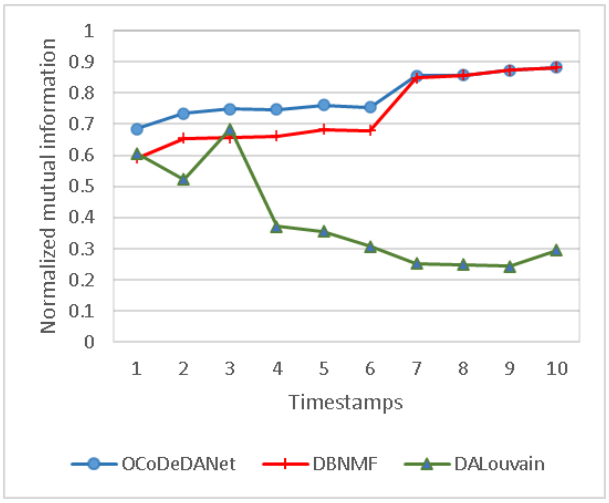
The results of the models for $p_1 = 0{,}21$ (Dataset 1), $q = 0{,}1$ (Dataset 2), and $r = 0{,}19$ (Dataset 3), $s = 1$ and $p_{intra} = 0{,}7$, are displayed in Fig. 2.13. OCoDeDANet outperforms DBNMF, and DBNMF outperforms DALouvain. From timestamp 7, OCoDeDANet and DBNMF are almost the same for the weakly assortative network and the disassortative network. At timestamps from 1 to 6, the presence of attributes is more relevant when the network is disassortative. The results of the models for $p_1 = 0{,}25$ (Dataset 4), $q = 0{,}1$ (Dataset 5), and $r = 0{,}19$ (Dataset 6), are omitted here for space purposes.

2.4.3. Experiments on overlapping synthetic networks

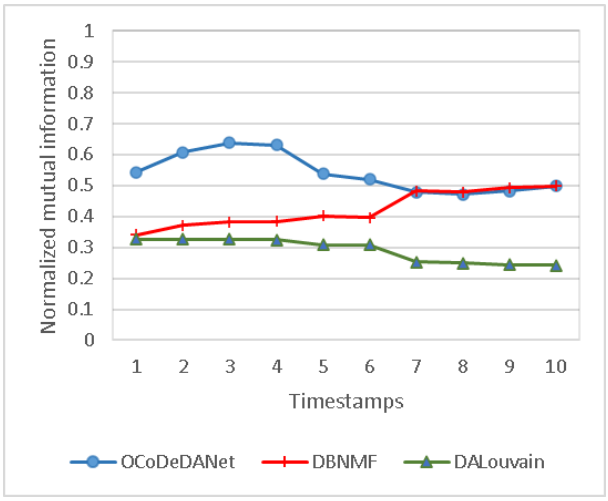
The performance of OCoDeDANet is tested on overlapping networks using Greene's benchmark (Greene et al., 2010). This benchmark addresses specific changes in communities of a dynamic network, such as birth and death, merge and split, expansion and contraction, and node switching between communities.



(a) Strongly assortative



(b) Weakly assortative



(c) Disassortative

Figure 2.13: Performance measured by NMI for Synthetic network 3, datasets 1, 2 and 3, and $s = 3$, $p_{intra} = 0{,}7$, $p_{inter} = 0{,}05$.

As in the case of disjoint networks, ten instances were generated using different random seeds for each synthetic network and parameter setting.

OCoDeDANet's parameters for the overlapping version are $\alpha = 0{,}8$, K equals the initial number of nodes of the processed network, and threshold $\delta = 0{,}3$. The other parameters are set as $a = 5$, $b = 3$, $MinIter = 200$, $MaxIter = 1000$. The parameters set for DBNMF's algorithm were the same. Both algorithms were executed with ten different random initializations of the non-negative matrices.

Attributes were added as described in Section 2.4.2. For the overlapping synthetic networks, considering $p_{inter} = 0{,}05$, the same configurations as in the case of disjoint synthetic networks are used: Case 1 is set as $s = 1$ and $p_{intra} = 0{,}7$, Case 2 is set as $s = 3$ and $p_{intra} = 0{,}7$, Case 3 is set as $s = 1$ and $p_{intra} = 1$, and Case 4 is set as $s = 3$ and $p_{intra} = 1$.

Next, we describe eleven overlapping synthetic networks and analyze the performance of the different algorithms in these networks.

Networks

Table 2.5 shows the parameters for the networks generated with this benchmark, along with a brief description for each one. In accordance with the specific parameters from Table 2.6, eleven datasets were created. For every dataset, five types were analyzed by altering the *muw* parameter, which can take on the values 0.1, 0.2, 0.3, 0.4, and 0.5. This parameter determines the level of clarity of the community structure within the generated network. A lower value suggests a strong community structure based on topology, while a higher value indicates a weak community structure.

Table 2.5: Parameter values that were used for Greene's benchmark. The description is shown as defined by Greene et al. (2010).

Parameter	Values	Description
N	250	Number of nodes
s	10	Number of time steps to generate
k	10	Average degree
$maxk$	20	Maximum degree
muw	0.1, 0.2, 0.3, 0.4, 0.5	Mixing parameter. Controls the overlap between communities
on	0, 5, 10, 25, 50	Number of overlapping nodes
om	2, 5, 10, 25	Number of memberships of the overlapping nodes
$birth$	2	Number of community birth events per time step
$death$	2	Number of community death events per time step
$merge$	2	Number of merge events per time step
$split$	2	Number of split events per time step
$expand$	5	Number of expansion events per time step
$contract$	5	Number of contraction events per time step
r	0.1	Rate of expansion/contraction
p	0.2	Probability of a node switching community membership between time steps

Table 2.6: Specific parameters for networks defined using Greene's benchmark.

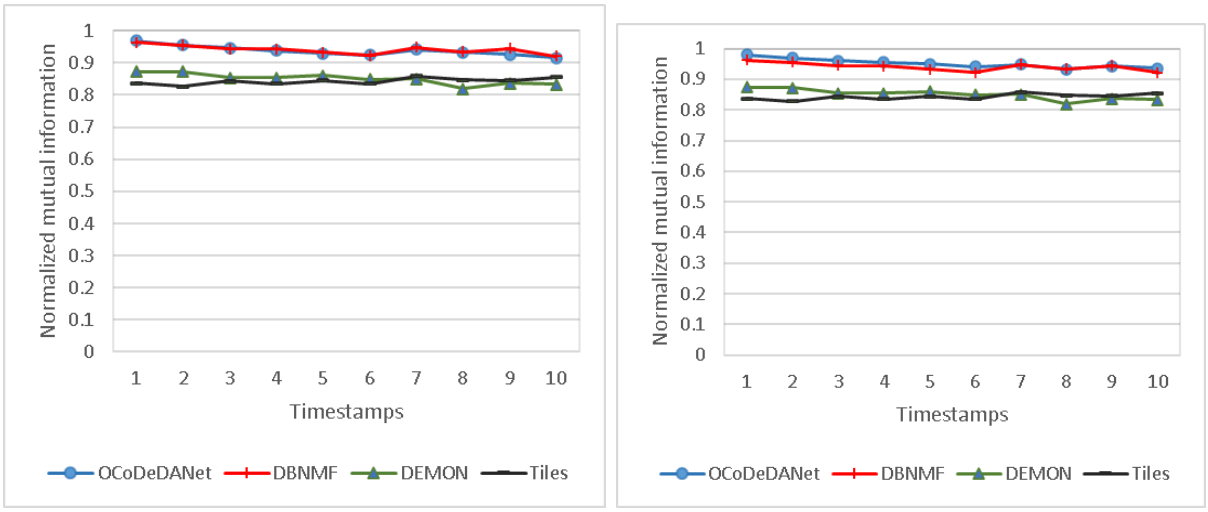
Dataset	Type	on	om
1	birth/death	0	-
2	birth/death	5	2
3	birth/death	10	2
4	birth/death	25	2
5	birth/death	50	2
6	birth/death	5	5
7	birth/death	5	10
8	birth/death	5	25
9	merge/split	5	2
10	expand/contract	5	2
11	switch	5	2

Results

Dataset 1

In Dataset 1, Type 3, there are no overlapping nodes, making it a disjoint network. The results of NMI for this network are shown in Fig. 2.14, where OCoDeDANet remain consistently

between 0.9 and 1 for all cases. For algorithms that do not take attributes into consideration for community detection, DBNMF displays a similar pattern to OCoDeDANet, while DEMON and Tiles have values between 0.8 and 0.9.



(a) Case 1, $s = 1$, $p_{intra} = 0{,}7$ (b) Case 4, $s = 3$, $p_{intra} = 1$

Figure 2.14: Performance for Dataset 1, Type 3 ($muw = 0{,}3$) of Overlapping synthetic network measured by NMI.

In Case 1, OCoDeDANet outperforms DBNMF by 1.1 %, DEMON by 11.9 % and Tiles by 12.9 %. In Case 4, DBNMF outperforms OCoDeDANet by 0.3 %, while OCoDeDANet outperforms DEMON by 10.3 % and Tiles by 11.3 %. These results indicate the importance of using relevant attributes to achieve better performance.

The results of NMI for Dataset 1, for the different values of muw tested, are presented in Table 2.7. When the network's topology is such that communities are more clear (Type 1, $muw = 0{,}1$), OCoDeDANet, DBNMF, and DEMON have the same NMI. However, as the community structure becomes less clear, using attributes results in a performance gain. In Case 4, OCoDeDANet outperforms DBNMF by 10.7 %, DEMON by 42.5 %, and Tiles by 45.3 %. In Case 1, OCoDeDANet still outperforms DBNMF by 1.9 %, DEMON by 31.1 %, and Tiles by 33.7 %.

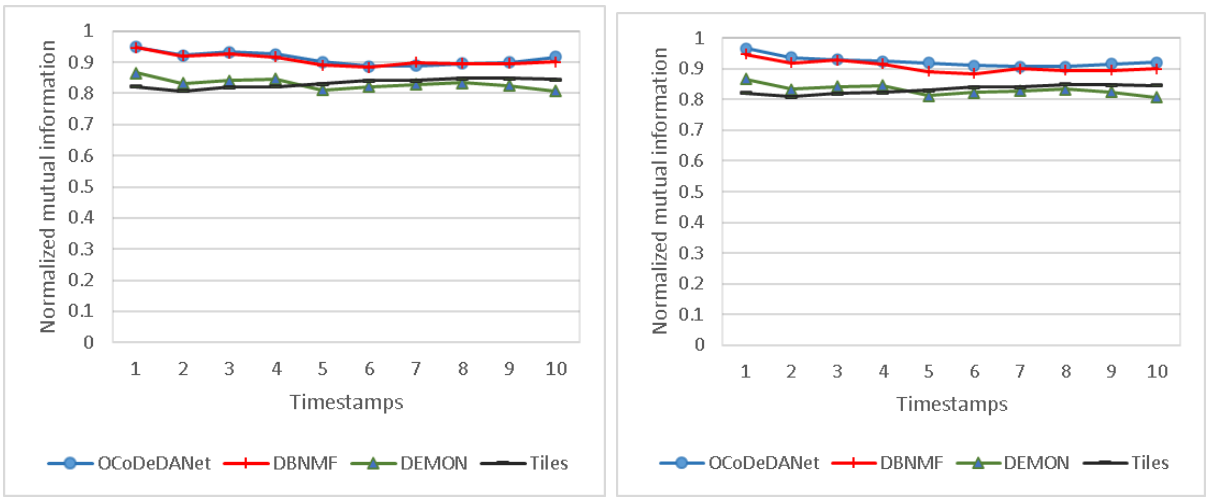
Table 2.7: Performance on dataset 1 from Greene's benchmark according to NMI. For OCoDeDANet, Case 4 ($s = 3$, $p_{intra} = 1$) to Case 1 ($s = 1$, $p_{intra} = 0{,}7$) are shown.

Network	OCoDeDANet				DBNMF	DEMON	Tiles
	Case 4	Case 3	Case 2	Case 1			
Dataset 1, Type 1	0.97	*0.98*	0.96	0.97	0.97	0.97	0.94
Dataset 1, Type 2	*0.97*	0.96	0.95	0.96	0.96	0.92	0.91
Dataset 1, Type 3	*0.95*	*0.95*	0.93	0.94	0.94	0.85	0.84
Dataset 1, Type 4	*0.92*	*0.92*	0.89	0.90	0.89	0.75	0.76
Dataset 1, Type 5	*0.90*	0.87	0.83	0.82	0.81	0.63	0.62

Datasets 2-5

Datasets 2 to 5 explore the algorithms' performance when the number of overlapping nodes is increased from 5 to 50 nodes.

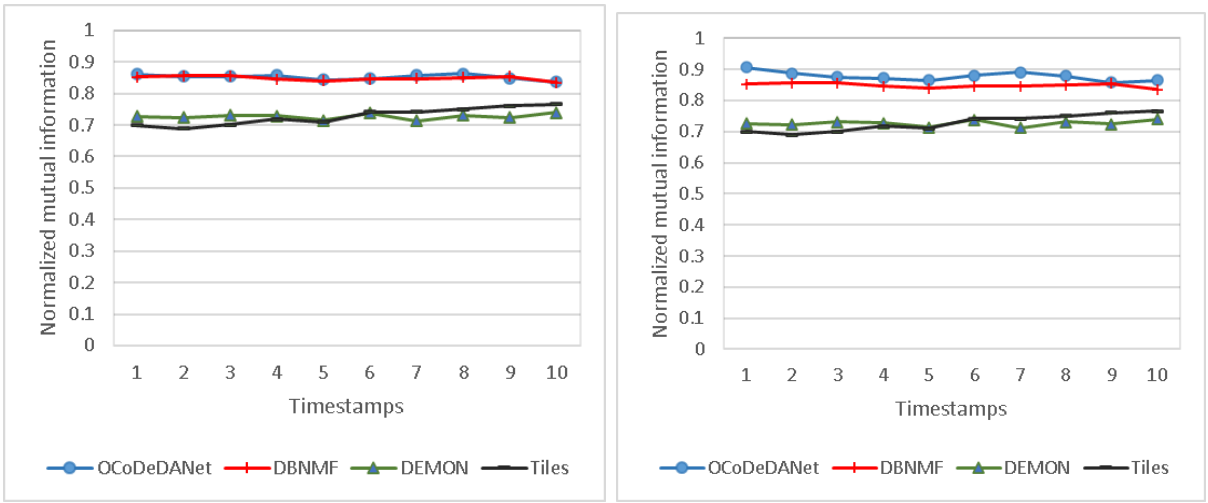
Fig. 2.15 shows the performance for Dataset 3, Type 3 according to NMI. OCoDeDANet outperforms DEMON and Tiles at every time step, with an approximate average of 10 %. Although OCoDeDANet outperforms DBNMF on average by 1.1 % for Case 1 and Case 4, OCoDeDANet is surpassed at time step 7.



(a) Case 1, $s = 1$, $p_{intra} = 0{,}7$ (b) Case 4, $s = 3$, $p_{intra} = 1$

Figure 2.15: Performance for Dataset 3, Type 3 ($muw = 0{,}3$) of Overlapping synthetic network measured by NMI.

Fig. 2.16 shows the performance according to NMI for Dataset 4, Type 4. OCoDeDANet outperforms DEMON and Tiles at every time step in this dataset by an approximate average of 19 %. Also, OCoDeDANet outperforms DBNMF on average by 2 % for Case 1 and Case 4.



(a) Case 1, $s = 1$, $p_{intra} = 0{,}7$ (b) Case 4, $s = 3$, $p_{intra} = 1$

Figure 2.16: Performance for Dataset 4, Type 4 ($muw = 0{,}4$) of Overlapping synthetic network measured by NMI.

The results of NMI for Dataset 2 to 5, for all five variations of muw are shown in Table 2.8. DEMON performs best when $muw = 0{,}1$, i.e., the community structure according

to topology is more prominent. When $muw = 0{,}2$, OCoDeDANet performs better than the other algorithms, but the difference with DBNMF is less than 1 %. As muw increases, all algorithms have a performance decrease, but in the case of OCoDeDANet the decrease is less, so the percentage gain of OCoDeDANet increases, reaching 15 % over DBNMF for Case 4.

Table 2.8: Performance on datasets 2 to 5 from Greene's benchmark according to NMI. For OCoDeDANet, Case 4 ($s = 3$, $p_{intra} = 1$) to Case 1 ($s = 1$, $p_{intra} = 0{,}7$) are shown.

Network	OCoDeDANet				DBNMF	DEMON	Tiles
	Case 4	Case 3	Case 2	Case 1			
Dataset 2, Type 1	*0.98*	0.97	0.97	0.97	0.97	0.97	0.93
Dataset 2, Type 2	*0.96*	*0.96*	0.94	0.95	0.95	0.91	0.9
Dataset 2, Type 3	*0.95*	0.94	0.92	0.93	0.93	0.85	0.84
Dataset 2, Type 4	*0.92*	0.91	0.88	0.89	0.89	0.75	0.75
Dataset 2, Type 5	*0.87*	0.83	0.8	0.78	0.76	0.62	0.62
Dataset 3, Type 1	0.95	*0.96*	0.95	0.95	*0.96*	*0.96*	0.93
Dataset 3, Type 2	*0.95*	*0.95*	0.94	*0.95*	0.94	0.91	0.9
Dataset 3, Type 3	*0.92*	*0.92*	0.9	0.91	0.91	0.83	0.83
Dataset 3, Type 4	0.9	*0.91*	0.87	0.88	0.88	0.74	0.74
Dataset 3, Type 5	*0.87*	0.82	0.8	0.78	0.76	0.61	0.61
Dataset 4, Type 1	0.93	0.93	0.91	0.92	0.93	*0.95*	0.9
Dataset 4, Type 2	0.91	*0.92*	0.9	0.91	0.91	0.9	0.87
Dataset 4, Type 3	*0.89*	*0.89*	0.86	0.88	*0.89*	0.82	0.81
Dataset 4, Type 4	*0.88*	0.87	0.84	0.85	0.85	0.73	0.73
Dataset 4, Type 5	*0.82*	0.78	0.73	0.73	0.71	0.6	0.6
Dataset 5, Type 1	0.88	0.9	0.87	0.89	0.89	*0.92*	0.88
Dataset 5, Type 2	0.86	*0.88*	0.84	0.87	*0.88*	0.87	0.84
Dataset 5, Type 3	0.84	*0.85*	0.81	0.83	0.84	0.8	0.79
Dataset 5, Type 4	0.8	*0.81*	0.75	0.78	0.78	0.71	0.7
Dataset 5, Type 5	*0.75*	0.71	0.67	0.66	0.64	0.58	0.57

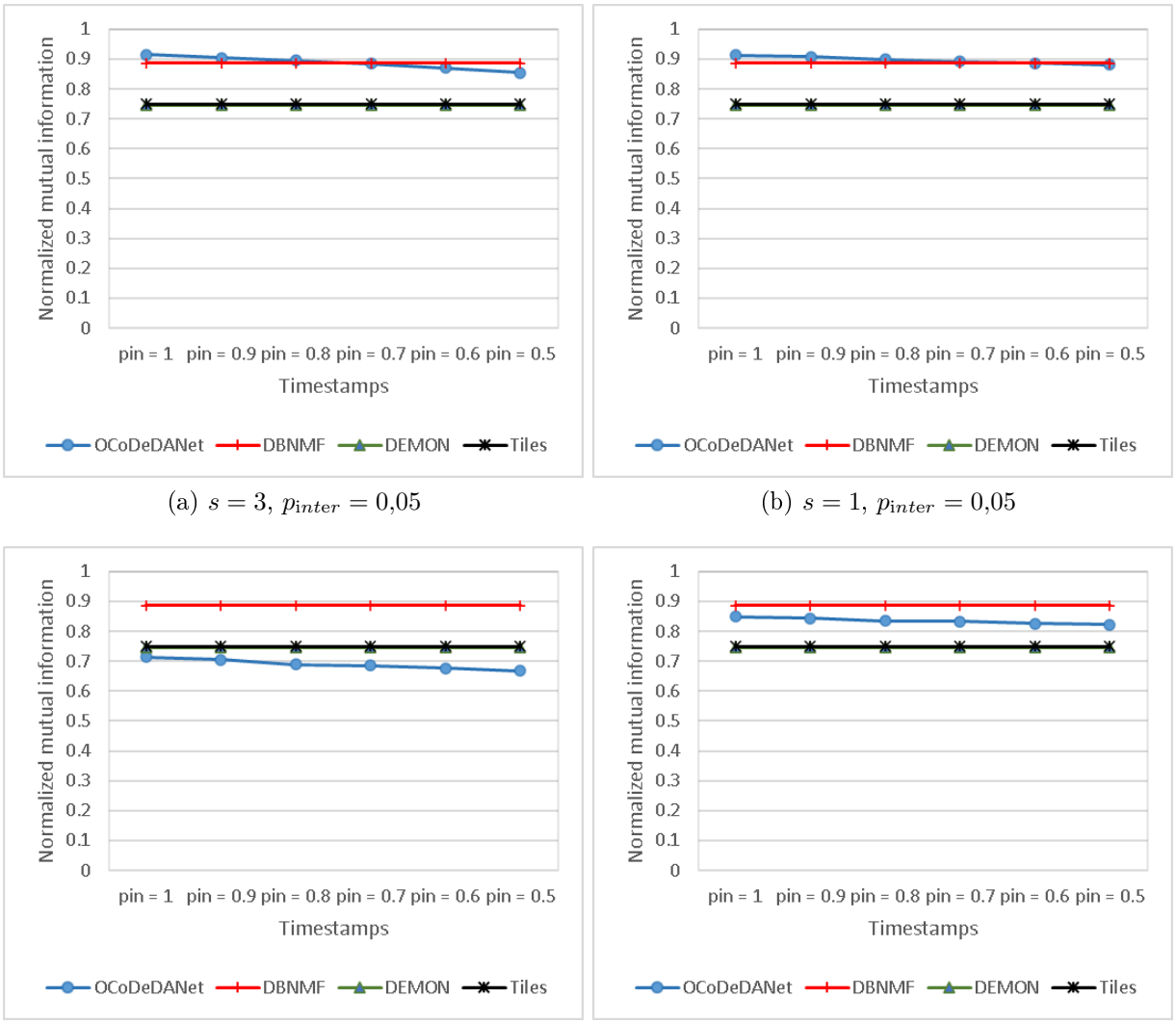
To conclude the analysis of these datasets, we study the results when p_{intra} and p_{inter} change over a broader range of values. These results are shown in Fig. 2.17. Comparing the results with the ones obtained in Fig. 2.5, they suggest that the overlapping problem is more challenging to solve than the disjoint one, at least for OCoDeDANet, with the attributes contributing less in the improvement of the results, and even causing worst results as compared with the case of not having any attributes. Furthermore, since the methodology for the generation of the attributes is the same, this will not work so thoroughly on the overlapping networks, and when $p_{inter} = 0{,}1$, having more attributes results in a worse performance than having fewer attributes, since they tend to confuse the algorithm instead of helping it.

Datasets 6-8

The datasets 6 to 8 analyze how algorithms perform as the membership of overlapping nodes increases from 5 to 25 communities.

Figure 2.18 illustrates the performance for Dataset 6, Type 5 based on NMI. OCoDeDANet consistently outperforms DEMON and Tiles at every time step, with an average improvement of approximately 28 %. However, although OCoDeDANet's average performance is 7 % better than DBNMF for Case 1 and Case 4, it is surpassed by DBNMF at timestamps 5 to 8.

The NMI results for Dataset 6 to 8, across all five variations of muw, are presented in Table 2.9. DEMON performs the best when $muw = 0{,}1$ and $muw = 0{,}2$, where the community structure according to topology is more prominent. When $muw = 0{,}3$ and $muw = 0{,}4$,



(a) $s = 3$, $p_{inter} = 0{,}05$

(b) $s = 1$, $p_{inter} = 0{,}05$

(c) $s = 3$, $p_{inter} = 0{,}1$

(d) $s = 1$, $p_{inter} = 0{,}1$

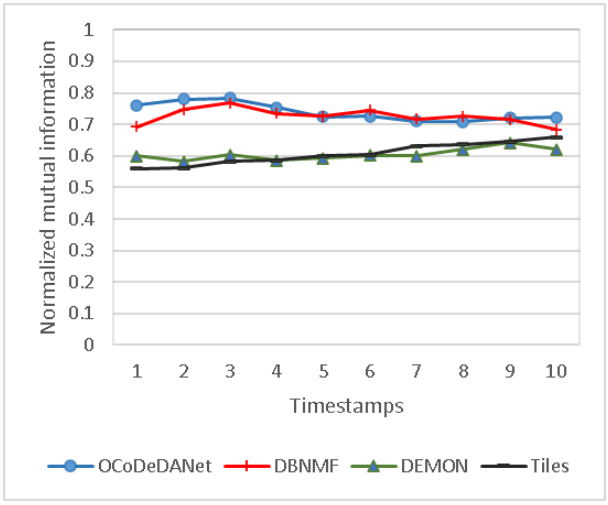
Figure 2.17: Performance for Dataset 2, Type 4, Overlapping synthetic network, measured by NMI, when p_{intra} is varied between 1 and 0.5.

OCoDeDANet outperforms DEMON and Tiles, and performs equally to DBNMF in Case 3. At $muw = 0{,}5$, OCoDeDANet outperforms all algorithms in Case 4 and Case 3, with a 5 % advantage over DBNMF, which is the closest algorithm in terms of NMI.

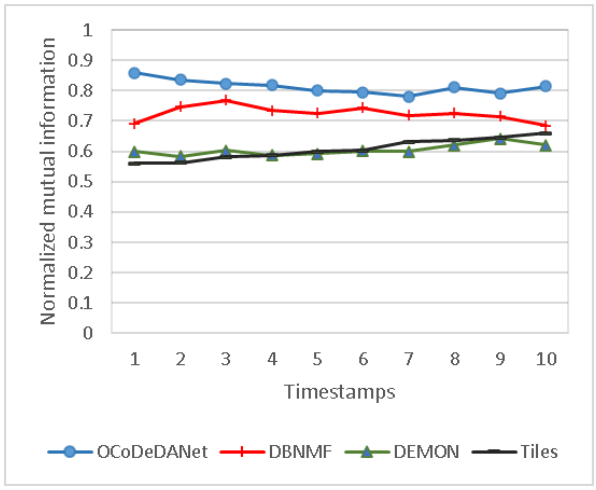
Dataset 9

Figure 2.19 shows the performance of Dataset 9, Type 1, based on NMI, when the communities of the network merge and split. DEMON outperforms the other three algorithms in all timestamps except time step 1. Furthermore, Tiles also outperforms OCoDeDANet and DBNMF. The lost on performance suggest that our algorithm is less effective in detecting communities than the other algorithms when $muw = 1$.

The NMI results for Dataset 9, across all five variations of muw, can be found in Table 2.10. As mentioned earlier, DEMON performs the best when $muw = 0{,}1$, which also



(a) Case 1, $s = 1$, $p_{intra} = 0{,}7$



(b) Case 4, $s = 3$, $p_{intra} = 1$

Figure 2.18: Performance for Dataset 6, Type 5 ($muw = 0{,}5$) of Overlapping synthetic network measured by NMI.

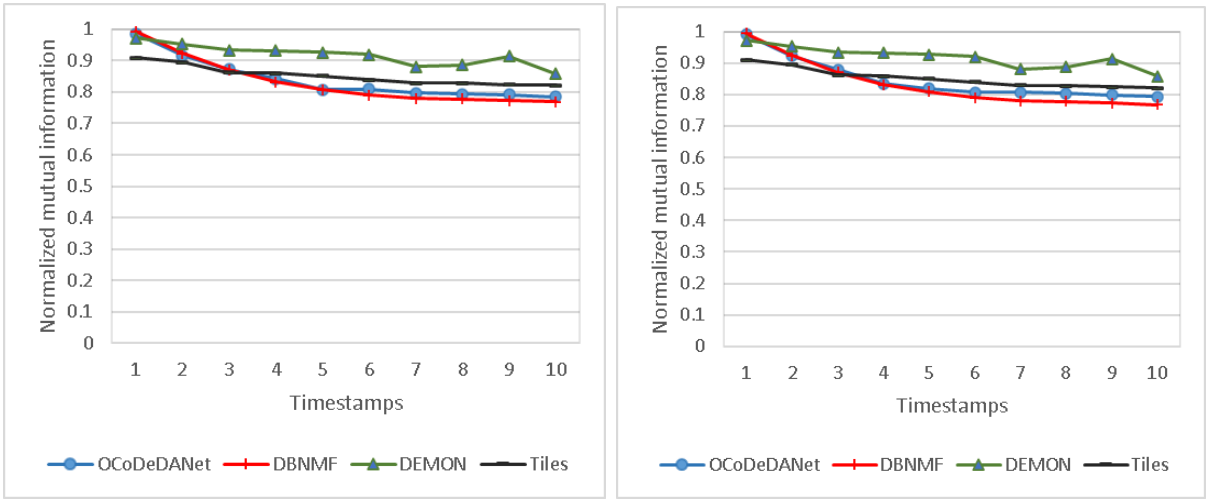
Table 2.9: Performance on datasets 6 to 8 from Greene's benchmark according to NMI. For OCoDeDANet, Case 4 ($s = 3$, $p_{intra} = 1$) to Case 1 ($s = 1$, $p_{intra} = 0{,}7$) are shown.

Network	OCoDeDANet				DBNMF	DEMON	Tiles
	Case 4	Case 3	Case 2	Case 1			
Dataset 6, Type 1	0.91	0.91	0.9	0.91	0.91	*0.95*	0.91
Dataset 6, Type 2	0.89	*0.9*	0.88	0.89	0.89	0.89	0.87
Dataset 6, Type 3	*0.89*	*0.89*	0.87	0.88	0.88	0.81	0.81
Dataset 6, Type 4	*0.85*	*0.85*	0.82	0.83	0.83	0.71	0.72
Dataset 6, Type 5	*0.81*	0.79	0.75	0.74	0.73	0.61	0.61
Dataset 7, Type 1	0.82	0.83	0.82	0.83	0.83	*0.89*	0.87
Dataset 7, Type 2	0.82	*0.83*	0.81	*0.83*	0.82	*0.83*	0.82
Dataset 7, Type 3	*0.79*	*0.79*	0.77	0.78	*0.79*	0.74	0.74
Dataset 7, Type 4	0.77	*0.78*	0.75	0.76	0.76	0.66	0.67
Dataset 7, Type 5	0.71	*0.72*	0.65	0.68	0.68	0.56	0.57
Dataset 8, Type 1	0.59	0.62	0.59	0.61	0.61	*0.67*	*0.67*
Dataset 8, Type 2	0.57	0.61	0.56	0.6	0.61	*0.64*	*0.64*
Dataset 8, Type 3	0.55	0.58	0.53	0.58	0.59	0.58	*0.6*
Dataset 8, Type 4	0.52	0.56	0.49	0.54	*0.58*	0.52	0.54
Dataset 8, Type 5	0.47	*0.49*	0.41	0.46	0.48	0.45	0.46

applies to $muw = 0{,}2$. When $muw = 0{,}3$, $muw = 0{,}4$, and $muw = 0{,}5$, OCoDeDANet on Case 4 outperforms all algorithms, with margins of 2.5 %, 6.6 %, and 22.6 % over its closest competitor, DBNMF, respectively.

Table 2.10: Performance on dataset 9 from Greene's benchmark according to NMI. For OCoDeDANet, Case 4 ($s = 3$, $p_{intra} = 1$) to Case 1 ($s = 1$, $p_{intra} = 0{,}7$) are shown.

Network	OCoDeDANet				DBNMF	DEMON	Tiles
	Case 4	Case 3	Case 2	Case 1			
Dataset 9, Type 1	0.85	0.85	0.84	0.84	0.83	*0.92*	0.85
Dataset 9, Type 2	0.85	0.85	0.84	0.84	0.84	*0.87*	0.82
Dataset 9, Type 3	*0.84*	*0.84*	0.82	0.83	0.82	0.79	0.75
Dataset 9, Type 4	*0.81*	0.8	0.78	0.77	0.76	0.68	0.66
Dataset 9, Type 5	*0.76*	0.73	0.69	0.67	0.62	0.55	0.53



(a) Case 1, $s = 1$, $p_{intra} = 0{,}7$ (b) Case 4, $s = 3$, $p_{intra} = 1$

Figure 2.19: Performance for Dataset 9, Type 1 ($muw = 0{,}1$) of Overlapping synthetic network measured by NMI.

Dataset 10

Dataset 10 explores the results when the network communities expand and contract.

Table 2.11 shows the NMI results for Dataset 10 across all five variations of muw. DBNMF performs best on $muw = 0{,}1$, OCoDeDANet performs the same as DBNMF when $muw = 0{,}2$, and for all the other parameters, OCoDeDANet on Case 4 outperforms all the other algorithms by 1 %, 2 %, and 10 % at least, when $muw = 0{,}3$, $muw = 0{,}4$ and $muw = 0{,}5$, respectively.

Table 2.11: Performance on dataset 10 from Greene's benchmark according to NMI. For OCoDeDANet, Case 4 ($s = 3$, $p_{intra} = 1$) to Case 1 ($s = 1$, $p_{intra} = 0{,}7$) are shown.

Network	OCoDeDANet				DBNMF	DEMON	Tiles
	Case 4	Case 3	Case 2	Case 1			
Dataset 10, Type 1	0.97	0.97	0.96	0.97	*0.98*	0.97	0.9
Dataset 10, Type 2	0.95	*0.96*	0.94	*0.96*	*0.96*	0.92	0.87
Dataset 10, Type 3	*0.95*	*0.95*	0.93	0.94	0.94	0.85	0.81
Dataset 10, Type 4	*0.92*	0.91	0.9	0.9	0.9	0.75	0.72
Dataset 10, Type 5	*0.88*	0.86	0.82	0.82	0.8	0.59	0.56

Dataset 11

Dataset 11 explores the results when the membership of nodes are switched.

Table 2.12 shows the NMI results for Dataset 11, across all five variations of muw. DEMON performs best on $muw = 0{,}1$, OCoDeDANet performs the same as DBNMF when $muw = 0{,}2$, OCoDeDANet on Case 4 outperforms all the other algorithms, by 1 %, 2 % and 8.8 % at least, when $muw = 0{,}3$, $muw = 0{,}4$ and $muw = 0{,}5$, respectively. OCoDeDANet on Case 3 surpassed Case 4 when $muw = 0{,}1$ and $muw = 0{,}4$.

Table 2.12: Performance on dataset 11 from Greene's benchmark according to NMI. For OCoDeDANet, Case 4 ($s = 3$, $p_{intra} = 1$) to Case 1 ($s = 1$, $p_{intra} = 0{,}7$) are shown.

Network	OCoDeDANet				DBNMF	DEMON	Tiles
	Case 4	Case 3	Case 2	Case 1			
Dataset 11, Type 1	0.95	0.96	0.95	0.96	0.96	*0.98*	0.9
Dataset 11, Type 2	*0.96*	*0.96*	0.94	0.95	*0.96*	0.93	0.87
Dataset 11, Type 3	*0.95*	*0.95*	0.93	0.94	0.94	0.86	0.81
Dataset 11, Type 4	0.91	*0.92*	0.88	0.9	0.89	0.75	0.7
Dataset 11, Type 5	*0.87*	0.86	0.82	0.82	0.8	0.59	0.54

2.4.4. Experiments on real-world networks

A crime network was designed to identify the relationships between those who have committed various property crimes in Chile between January 1st, 2019 and December 31st, 2021. Property crimes such as *robbery in an uninhabited place*, *robbery with violence*, and *homicide in a fight* fall under the scope of this network (Márquez & Weber, 2023).

Network

The crime network under consideration is comprised of 12 time steps, each of which corresponds to a 3-month period based on the date of the crime. It starts with 38 nodes and 39 edges, and concludes with 419 nodes and 2087 edges. The network contains no vertex deletions. Each node in the network represents an individual who has been charged with a crime, while an edge between two nodes indicates that those individuals have been charged with the same crime. The network possesses a set of attributes $\boldsymbol{F}^{(t)}$, where a value of 1 for $F_{nm}^{(t)}$ signifies that individual n has committed crime type m up to time step t, where m corresponds to one of the 26 crime types included in the network (Márquez & Weber, 2023).

Results

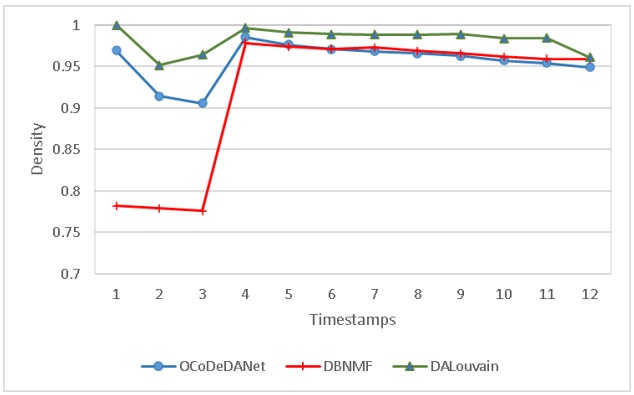
The performance of OCoDeDANet and other models is displayed in Fig. 2.20. DALouvain produced the best density and modularity results, followed by OCoDeDANet and DBNMF. On the other hand, when it comes to entropy, OCoDeDANet showed the best results, followed closely by DBNMF and then DALouvain. This suggests that OCoDeDANet prioritizes attribute importance while DALouvain places more emphasis on topology.

However, since neither metric provides a clear advantage to any algorithm, no direct conclusions can be drawn. In Section 2.4.5.3, we follow the proposed approach from Márquez & Weber (2023) to address this limitation.

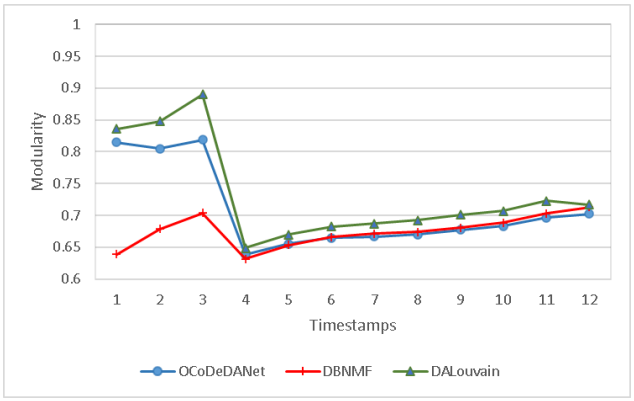
To gain a better understanding of the results of OCoDeDANet, Fig. 2.21 displays the three communities with a higher number of members at time step 12.

The third largest community, which consists of 30 nodes and is displayed on the left part of Fig. 2.21, has three connected components of 21, 6 and 3 nodes, respectively. Within this community, there are 47 edges connecting the nodes. The attribute values enable the nodes to be assigned to this community.

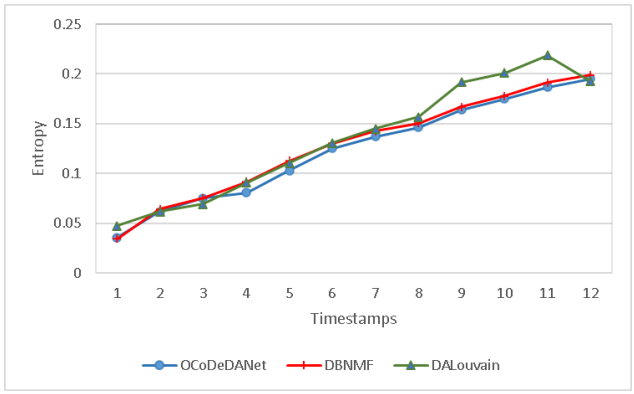
The second largest community, comprised of 45 nodes, is located in the center of Fig. 2.21.



(a) Density



(b) Modularity



(c) Entropy

Figure 2.20: Performance for the crime network.

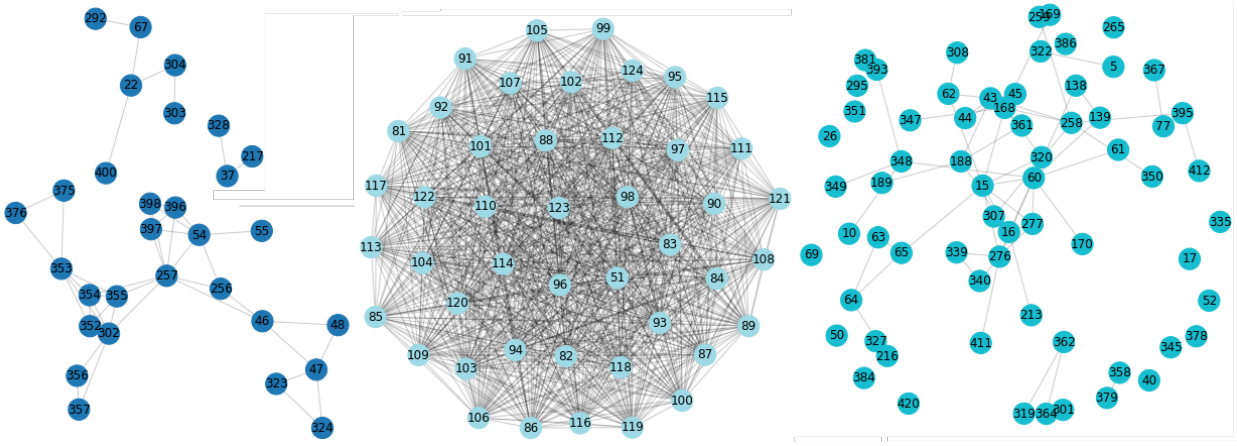


Figure 2.21: Communities obtained by OCoDeDANet at $t = 12$ for the crime network.

This community displays a denser connection between its nodes than the other two communities in the figure, having 981 edges connecting these nodes.

The largest community, composed of 66 nodes and shown on the right side of Fig. 2.21, has only 57 edges joining these nodes, and several unconnected nodes within the community. However, the attributes values enforce this nodes to be part of the community.

The community shown on the right side of Fig. 2.21 is the largest one, consisting of 66 nodes. Despite having only 57 edges and several unconnected nodes within the community, the attribute values of these nodes enforce them to be considered as part of the community.

2.4.5. Summary of results

2.4.5.1. Disjoint synthetic networks

In previous results, we compare the performance of different disjoint algorithms on the synthetic attributed networks over time, for every dataset. We will now present a comparison of these algorithms based on NMI, by averaging the results of all time steps and all seeds for each model. This comparison is illustrated in Table 2.13.

Table 2.13: Performance on disjoint synthetic networks according to NMI. For OCoDeDANet and DALouvain, Case 4 ($s = 3$, $p_{intra} = 1$) and Case 1 ($s = 1$, $p_{intra} = 0{,}7$) are shown.

Network	OCoDeDANet		DALouvain		DBNMF
	Case 4	Case 1	Case 4	Case 1	
Syn. net. 1, Dataset 1	0.8	0.62	0.57	0.51	0.55
Syn. net. 1, Dataset 2	0.86	0.67	0.58	0.56	0.59
Syn. net. 1, Dataset 3, Fig. 2.2a	0.81	0.62	0.53	0.51	0.54
Syn. net. 1, Dataset 4, Fig. 2.2b	0.81	0.6	0.48	0.47	0.52
Syn. net. 1, Dataset 5	0.77	0.58	0.46	0.44	0.52
Syn. net. 1, Dataset 6	0.86	0.66	0.54	0.52	0.58
Syn. net. 1, Dataset 7, Fig. 2.2c	0.8	0.6	0.49	0.48	0.53
Syn. net. 1, Dataset 8, Fig. 2.2d	0.84	0.64	0.5	0.48	0.56
Syn. net. 1, Dataset 9	0.84	0.65	0.56	0.54	0.58
Syn. net. 1, Dataset 10	0.9	0.69	0.55	0.54	0.61
Syn. net. 1, Dataset 11, Fig. 2.2e	0.84	0.64	0.51	0.49	0.56
Syn. net. 1, Dataset 12, Fig. 2.2f	0.85	0.66	0.55	0.52	0.59
Syn. net. 2, Dataset 1, Fig. 2.8a	0.85	0.79	0.57	0.39	0.79
Syn. net. 2, Dataset 2, Fig. 2.9a	0.86	0.77	0.31	0.25	0.73
Syn. net. 3, Dataset 1, Fig. 2.13a	0.98	0.88	0.74	0.36	0.83
Syn. net. 3, Dataset 2, Fig. 2.13b	0.97	0.79	0.78	0.39	0.74
Syn. net. 3, Dataset 3, Fig. 2.13c	0.86	0.54	0.29	0.29	0.42
Syn. net. 3, Dataset 4	0.98	0.95	0.74	0.45	0.93
Syn. net. 3, Dataset 5	0.96	0.78	0.59	0.31	0.72
Syn. net. 3, Dataset 6	0.93	0.78	0.29	0.29	0.64

Based on the average Normalized Mutual Information (NMI), OCoDeDANet performed better than the other models that were compared. Among all the test networks, OCoDeDANet in Case 4 had the highest NMI of 0.869, followed by OCoDeDANet in Case 1 with 0.696, DBNMF with 0.627, DALouvain in Case 4 with 0.532 and DALouvain in Case 1 with 0.44.

Therefore, our proposed model outperformed DALouvain in terms of NMI, which also takes into account both attribute values and topology, by more than 58 % considering both Case 4 and Case 1. Additionally, it also performed better than DBNMF which considers only the dynamic aspect without attributes, by 38.6 % in Case 4 and by 11 % in Case 1.

2.4.5.2. Overlapping synthetic networks

In this summary, we compare the overlapping algorithms based on NMI by averaging the results of all time steps, all seeds, and all *muw* values for each model. This comparison is shown in Table 2.14.

Table 2.14: Performance on Overlapping synthetic networks according to NMI. For OCoDeDANet, Case 4 ($s = 3$, $p_{intra} = 1$) to Case 1 ($s = 1$, $p_{intra} = 0{,}7$) are shown.

Network	OCoDeDANet				DBNMF	DEMON	Tiles
	Case 4	Case 3	Case 2	Case 1			
Dataset 1, Fig. 2.14	0.94	0.94	0.91	0.92	0.91	0.82	0.81
Dataset 2	0.94	0.92	0.90	0.90	0.90	0.82	0.81
Dataset 3, Fig. 2.15	0.92	0.91	0.89	0.89	0.89	0.81	0.80
Dataset 4, Fig. 2.16	0.89	0.88	0.85	0.86	0.86	0.80	0.78
Dataset 5	0.83	0.83	0.79	0.81	0.81	0.78	0.76
Dataset 6, Fig. 2.18	0.87	0.87	0.84	0.85	0.85	0.79	0.78
Dataset 7	0.78	0.79	0.76	0.78	0.78	0.74	0.73
Dataset 8	0.54	0.57	0.52	0.56	0.57	0.57	0.58
Dataset 9, Fig. 2.19	0.82	0.81	0.79	0.79	0.77	0.76	0.72
Dataset 10	0.93	0.93	0.91	0.92	0.92	0.82	0.77
Dataset 11	0.93	0.93	0.9	0.91	0.91	0.82	0.76

Based on the average NMI, OCoDeDANet outperformed the other compared models. Among all the test networks, OCoDeDANet in Case 4 had the highest NMI of 0.854, followed by OCoDeDANet in Case 3 with 0.853, OCoDeDANet in Case 1 with 0.835, DBNMF with 0.834, OCoDeDANet in Case 2 with 0.824, DEMON with 0.775 and Tiles with 0.755. Therefore, our proposed model outperformed DBNMF by 2.4 %, DEMON by 10.1 % and Tiles by 13.1 % when attributes clearly contributed to identifying communities. For cases with $p_{intra} = 0{,}7$ the results suggest that adding more attributes caused a decrease in performance.

2.4.5.3. Real-world networks

We can compare algorithms by normalizing each metric and computing the average if there is no clear advantage of an algorithm when including both metrics. However, since the scales of these metrics vary significantly, we decided to compare the algorithms on a one-on-one basis. This pairwise comparison involves measuring the percentage difference in density and entropy and then averaging them (Márquez & Weber, 2023). We also determine the average percentage difference using modularity and entropy.

Table 2.15 displays the results for the crime network, and it reveals that OCoDeDANet performed the best among the compared algorithms. When including density and entropy, it outperformed DBNMF by 3.6 %, and when including modularity and entropy, it outperformed DBNMF by 4 %. Furthermore, it outperformed DALouvain by 2.7 % when including density and entropy, and by 2.3 % when including modularity and entropy. In terms of topology metric, DALouvain performed the best, followed by OCoDeDANet and DBNMF. According to the attribute metric, OCoDeDANet performed the best, followed by DBNMF and DALouvain. This suggests that OCoDeDANet is mainly building its communities based on their attributes.

Table 2.15: Performance on the crime network according to modularity, density and entropy.

	Density	Modularity	Entropy
OCoDeDANet	0.957	0.708	0.124
DBNMF	0.921	0.675	0.128
OCoDeDANet *vs* DBNMF (%)	3.90	4.80	3.26
DALouvain	0.982	0.734	0.135
OCoDeDANet *vs* DALouvain (%)	-2.62	-3.54	8.04

2.5. Concluding remarks and future work

Real-world networks require methods that capture their evolving essence and complexity. However, the study of dynamic attributed networks has primarily focused on static information including attributes, or dynamic information focusing solely on the graph's links. In response to these limitations, we have introduced the OCoDeDANet algorithm. This novel approach for overlapping community detection in dynamic attributed networks processes changes in the graph's structure and the attributes of its nodes over time. The method's use of probabilistic non-negative matrix factorization widens its scope by capturing information on both topology and attributes, including past information. Additionally, the automatic determination of the number of communities broadens its applications.

We tested our algorithm on different disjoint and overlapping synthetic networks, as well as a real-world network. Normalized mutual information, in its disjoint and overlapping form, was employed to address the models' performance on networks with ground truth. Density and entropy were used in the absence of ground truth. The results showed that our approach outperforms other state-of-the-art methods.

The performance of our algorithm indicates that considering attributes improves community detection. Nevertheless, there are opportunities for further improvement. The algorithm would benefit from automatically calibrating parameters such as the dynamic matrices' weights and the threshold for overlapping community detection.

Another aspect that can be improved is handling irrelevant attributes within the model. If, for any reason, some of the attributes used are not genuinely relevant to detecting communities, the model would still consider the respective information, potentially leading to skewed results. Therefore, the model's accuracy and reliability can benefit from adding weights to leverage the impact of topology and attributes.

Finally, the algorithm's applicability can be enhanced by testing the approach for overlapping community detection, including membership degrees for each group. In this case, the result would allow us to determine the degree of participation of each node in cases where it belongs to multiple communities.

Conclusion

This thesis addresses some limitations of community detection approaches in dynamic attributed networks, which often fail to incorporate evolving network structure and changing node attributes comprehensively. To overcome this gap, we propose two novel algorithms, CoDeDANet and OCoDeDANet, designed to adapt to real-world network dynamics by jointly capturing structural and attribute changes over time. Both models demonstrate superior performance compared to state-of-the-art methods across various synthetic and real-world dynamic networks by integrating topological and attribute information.

CoDeDANet focuses on detecting disjoint communities, utilizing spectral clustering combined with tensor methods to incorporate current and historical network data in an algorithm that needs the number of communities as input. OCoDeDANet improves this approach by detecting overlapping communities through probabilistic non-negative matrix factorization, using automatic relevance determination to infer the number of communities. Both algorithms consider the benefits of including node attributes in improving the accuracy of community detection, particularly as network structures in evolving contexts become more complex.

We evaluated both algorithms using synthetic and real-world datasets, applying standard metrics such as normalized mutual information to networks with known ground truths and measures such as density and entropy for networks without ground truth. For OCoDeDANet, overlapping normalized mutual information was also used when overlapping structures were present on the networks. The results showed that CoDeDANet and OCoDeDANet consistently outperform competing state-of-the-art methods, especially in scenarios where node attributes are relevant to detect communities. Additionally, we observed that OCoDeDANet performed better in cases where the topological structure of the network alone is insufficient to capture the community dynamics, further highlighting the importance of including node attributes in the analysis.

Despite its advantages, CoDeDANet and OCoDeDANet present some opportunities for improvement. Both algorithms assume that all attributes provided are relevant to the community detection task, needing a mechanism for discerning which attributes are relevant and which may distort the results, which could lead to suboptimal results if irrelevant attributes are included. Future work could focus on incorporating a procedure that automatically filters out irrelevant attributes, assigning weights to attributes based on their relevance to the community structure, thus improving the algorithm's robustness.

Both models could benefit from computational optimizations to improve scalability since the size of the networks that can be processed is limited. For CoDeDANet, only the eigenva-

lues affected by changes could be updated, avoiding computing the whole matrix on each time step. For OCoDeDANet, alternatives such as using dimension reduction techniques applied to the obtained matrices or block-wise updates can allow processing of larger networks.

Another aspect to improve for CoDeDANet is that it currently requires the number of communities to be input manually, which could limit its practical applicability in cases where this information is unknown. Developing an automated mechanism to determine the number of communities is crucial for future research. For the case of OCoDeDANet, the handling of overlapping communities could be extended by introducing membership degrees for nodes, allowing for a more granular understanding of a node's participation in multiple communities.

In conclusion, this thesis contributes two new models that improve the detection of communities in dynamic attributed networks, addressing both disjoint and overlapping communities. These models offer new insights into the structure of dynamic networks and provide tools for analyzing the complex interactions and evolving behaviors that characterize real-world systems. By integrating topological and attribute-based information, the presented algorithms enhance the ability to detect communities more comprehensively and realistically, making them valuable for applications in social network analysis, biological systems, technological networks, and beyond.

Bibliography

Abdrabbah, S. B., Ayachi, R., & Amor, N. B. (2014). Collaborative filtering based on dynamic community detection. *Proceedings of the 2nd International Conference on Dynamic Networks and Knowledge Discovery*, 1229, 85–106. doi:10.5555/3053802.3053813.

Afsariardchi, N. (2012). Community Detection in Dynamic Networks. M.S. thesis, Department of Electrical and Computer Engineering, McGill University, Montreal, Canada. url:https://escholarship.mcgill.ca.

Al-sharoa, E., Al-khassaweneh, M., & Aviyente, S. (2019). Tensor Based Temporal and Multilayer Community Detection for Studying Brain Dynamics During Resting State fMRI. *IEEE Transactions on Biomedical Engineering*, 66(3), 695–709. doi:10.1109/TBME.2018.2854676.

Amelio, A. & Pizzuti, C. (2014). Overlapping Community Discovery Methods: A Survey. In A. Gunduz-Oguducu & A. Etaner-Uyar (Eds.), *Social Networks: Analysis and Case Studies. Lecture Notes in Social Networks* (pp. 105–125). Vienna: Springer. doi:10.1007/978-3-7091-1797-2_6.

Arbelaitz, O., Gurrutxaga, I., Muguerza, J., Pérez, J., & Perona, I. (2013). An extensive comparative study of cluster validity indices. *Pattern Recognition*, 46(1), 243–256. doi:10.1016/j.patcog.2012.07.021.

Atay, Y., Koc, I., Babaoglu, I., & Kodaz, H. (2017). Community detection from biological and social networks: A comparative analysis of metaheuristic algorithms. *Applied Soft Computing*, 50, 194–211. doi:10.1016/j.asoc.2016.11.025.

Bahadori, S., Zare, H., & Moradi, P. (2021). PODCD: Probabilistic overlapping dynamic community detection. *Expert Systems with Applications*, 174, Article 114650. doi:10.1016/j.eswa.2021.114650.

Bedi, P. & Sharma, C. (2016). Community detection in social networks. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 6(3), 115–135. doi:10.1002/widm.1178.

Bello, G. A., Harenberg, S., Agrawal, A., & Samatova, N. F. (2016). Community detection in dynamic attributed graphs. In *International Conference on Advanced Data Mining and Applications* (pp. 329–344). doi:10.1007/978-3-319-49586-6_22.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., & Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10), Article P10008. doi:10.1088/1742-5468/2008/10/p10008.

Bothorel, C., Cruz, J. D., Magnani, M., & Micenkova, B. (2015). Clustering attributed graphs: models, measures and methods. *Network Science*, 3(3), 408–444. doi:10.1017/nws.2015.9.

Cao, J., Jin, D., Yang, L., & Dang, J. (2018). Incorporating network structure with node contents for community detection on large networks using deep learning. *Neurocomputing*, 297, 71–81. doi:10.1016/j.neucom.2018.01.065.

Cazabet, R. & Rossetti, G. (2023). Challenges in community discovery on temporal networks. In P. Holme & J. Saramäki (Eds.), *Temporal Network Theory* (pp. 185–202). Cham: Springer International Publishing. doi:10.1007/978-3-031-30399-9_10.

Chakraborty, T. & Chakraborty, A. (2013). OverCite: Finding overlapping communities in citation network. In *2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2013)* (pp. 1124–1131). doi:10.1145/2492517.2500255.

Chakraborty, T., Dalmia, A., Mukherjee, A., & Ganguly, N. (2017). Metrics for community analysis: A survey. *ACM Computing Surveys (CSUR)*, 50(4), 1–37. doi:10.1145/3091106.

Chang, Z., Jia, C., Yin, X., & Zheng, Y. (2019). A generative model for exploring structure regularities in attributed networks. *Information Sciences*, 505, 252–264. doi:10.1016/j.ins.2019.07.084.

Cheng, F., Cui, T., Su, Y., Niu, Y., & Zhang, X. (2018). A local information based multiobjective evolutionary algorithm for community detection in complex networks. *Applied Soft Computing*, 69, 357–367. doi:10.1016/j.asoc.2018.04.037.

Chunaev, P. (2020). Community detection in node-attributed social networks: a survey. *Computer Science Review*, 37, 100286. doi:10.1016/j.cosrev.2020.100286.

Combe, D., Largeron, C., Géry, M., & Egyed-Zsigmond, E. (2015). I-louvain: An attributed graph clustering method. In E. Fromont, T. De Bie, & M. van Leeuwen (Eds.), *Advances in Intelligent Data Analysis XIV. IDA 2015. Lecture Notes in Computer Science* (pp. 181–192). doi:10.1007/978-3-319-24465-5_16.

Coscia, M., Rossetti, G., Giannotti, F., & Pedreschi, D. (2012). DEMON: a local-first discovery method for overlapping communities. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 615–623). doi:10.1145/2339530.2339630.

Dang, T. A. & Viennet, E. (2012). Community detection based on structural and attribute similarities. *The Sixth International conference on digital society (ICDS 2012)*, 659, 7–12. url:marami-2011.imag.fr.

Danon, L., Diaz-Guilera, A., Duch, J., & Arenas, A. (2005). Comparing community structure

identification. *Journal of statistical mechanics: Theory and experiment*, 2005(09), Article P09008. doi:10.1088/1742-5468/2005/09/P09008.

Ferrara, E. (2012). Community structure discovery in facebook. *International Journal of Social Network Mining*, 1(1), 67–90. doi:10.1504/IJSNM.2012.045106.

Ferrara, E., De Meo, P., Catanese, S., & Fiumara, G. (2014). Detecting criminal organizations in mobile phone networks. *Expert Systems with Applications*, 41(13), 5733–5750. doi:10.1016/j.eswa.2014.03.024.

Fortunato, S. & Hric, D. (2016). Community detection in networks: A user guide. *Physics Reports*, 659, 1–44. doi:10.1016/j.physrep.2016.09.002.

Gao, Y., Yu, X., & Zhang, H. (2020). Uncovering overlapping community structure in static and dynamic networks. *Knowledge-Based Systems*, 201, 106060. doi:10.1016/j.knosys.2020.106060.

Girvan, M. & Newman, M. E. J. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences of the United States of America*, 99(12), 7821–7826. doi:10.1073/pnas.122653799.

Greene, D., Doyle, D., & Cunningham, P. (2010). Tracking the evolution of communities in dynamic social networks. In *2010 International Conference on Advances in Social Networks Analysis and Mining* (pp. 176–183). doi:10.1109/ASONAM.2010.17.

He, C., Fei, X., Cheng, Q., Li, H., Hu, Z., & Tang, Y. (2022). A survey of community detection in complex networks using nonnegative matrix factorization. *IEEE Transactions on Computational Social Systems*, 9(2), 440–457. doi:10.1109/TCSS.2021.3114419.

Huang, H., Wang, X., & Yu, G. (2018). Community detection based on unified bayesian nonnegative matrix factorization. In *2018 IEEE 3rd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)* (pp. 395–403). doi:10.1109/ICCCBDA.2018.8386549.

Huang, Z., Zhong, X., Wang, Q., Gong, M., & Ma, X. (2020). Detecting community in attributed networks by dynamically exploring node attributes and topological structure. *Knowledge-Based Systems*, 196, Article 105760. doi:10.1016/j.knosys.2020.105760.

Jin, D., Yu, Z., Jiao, P., Pan, S., He, D., Wu, J., Yu, P. S., & Zhang, W. (2023). A survey of community detection approaches: From statistical modeling to deep learning. *IEEE Transactions on Knowledge and Data Engineering*, 35(2), 1149–1170. doi:10.1109/TKDE.2021.3104155.

Khanam, K. Z., Srivastava, G., & Mago, V. (2022). The homophily principle in social network analysis: A survey. *Multimedia Tools and Applications*, (pp. 1–44). doi:10.1007/s11042-021-11857-1.

Kolda, T. G. & Bader, B. W. (2009). Tensor Decompositions and Applications. *SIAM Review*, 51(3), 455–500. doi:10.1137/07070111X.

Lalwani, D., Somayajulu, D. V., & Krishna, P. R. (2015). A community driven social recommendation system. *2015 IEEE International conference on big data (big data)*, (pp. 821–826). doi:10.1109/BigData.2015.7363828.

Lancichinetti, A., Fortunato, S., & Kertész, J. (2009). Detecting the overlapping and hierarchical community structure in complex networks. *New Journal of Physics*, 11(3), 033015. doi:10.1088/1367-2630/11/3/033015.

Largeron, C., Mougel, P.-N., Benyahia, O., & Zaïane, O. R. (2017). DANCer: dynamic attributed networks with community structure generation. *Knowledge and Information Systems*, 53(1), 109–151. doi:10.1007/s10115-017-1028-2.

Lee, D. & Seung, H. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, 401, 788–791. doi:10.1038/44565.

Lee, D. & Seung, H. (2001). Algorithms for non-negative matrix factorization. *Advances in Neural Information Processing Systems 13 (NIPS 2000)*, 13(1), 556–562. url:proceedings.neurips.cc.

Li, W., Zhou, X., Yang, C., Fan, Y., Wang, Z., & Liu, Y. (2022). Multi-objective optimization algorithm based on characteristics fusion of dynamic social networks for community discovery. *Information Fusion*, 79, 110–123. doi:10.1016/j.inffus.2021.10.002.

Li, W., Zhu, H., Li, S., Wang, H., Dai, H., Wang, C., & Jin, Q. (2021). Evolutionary community discovery in dynamic social networks via resistance distance. *Expert Systems with Applications*, 171, Article 114536. doi:10.1016/j.eswa.2020.114536.

Lu, H., Sang, X., Zhao, Q., & Lu, J. (2020). Community detection algorithm based on nonnegative matrix factorization and pairwise constraints. *Physica A: Statistical Mechanics and its Applications*, 545, Article 123491. doi:10.1016/j.physa.2019.123491.

Ma, H., Liu, Z., Zhang, X., Zhang, L., & Jiang, H. (2021). Balancing topology structure and node attribute in evolutionary multi-objective community detection for attributed networks. *Knowledge-Based Systems*, 227, Article 107169. doi:10.1016/j.knosys.2021.107169.

Ma, L., Gong, M., Liu, J., Cai, Q., & Jiao, L. (2014). Multi-level learning based memetic algorithm for community detection. *Applied Soft Computing*, 19, 121–133. doi:10.1016/j.asoc.2014.02.003.

Ma, X. & Dong, D. (2017). Evolutionary Nonnegative Matrix Factorization Algorithms for Community Detection in Dynamic Networks. *IEEE Transactions on Knowledge and Data Engineering*, 29(5), 1045–1058. doi:10.1109/TKDE.2017.2657752.

Mahmoud, H., Masulli, F., Rovetta, S., & Russo, G. (2014). Community detection in protein-protein interaction networks using spectral and graph approaches. In *Formenti E., Tagliaferri R., Wit E. (eds) Computational Intelligence Methods for Bioinformatics and Biostatistics. CIBB 2013. Lecture Notes in Computer Science*, volume 8452 (pp. 62–75). doi:10.1007/978-3-319-09042-9_5.

Mankad, S. & Michailidis, G. (2013). Structural and functional discovery in dynamic networks with non-negative matrix factorization. *Physical Review E*, 88(4), 042812. doi:10.1103/PhysRevE.88.042812.

Márquez, R., Weber, R., & De Carvalho, A. C. (2019). A non-negative matrix factorization approach to update communities in temporal networks using node features. In *2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)* (pp. 728–732). doi:10.1145/3341161.3343677.

Moradi-Jamei, B., Kramer, B. L., Calderón, J. B. S., & Korkmaz, G. (2021). Community formation and detection on github collaboration networks. In *Proceedings of the 2021 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining* (pp. 244–251). doi:10.1145/3487351.3488278.

Márquez, R. & Weber, R. (2023). Dynamic community detection including node attributes. *Expert Systems with Applications*, 223, 119791. doi:10.1016/j.eswa.2023.119791.

Nath, K., Roy, S., & Nandi, S. (2020). InOvIn: A fuzzy-rough approach for detecting overlapping communities with intrinsic structures in evolving networks. *Applied Soft Computing*, 89, Article 106096. doi:10.1016/j.asoc.2020.106096.

Newman, M. E. J. (2006). Modularity and community structure in networks. *Proceedings of the National Academy of Sciences of the United States of America*, 103(23), 8577–8582. doi:10.1073/pnas.0601602103.

Newman, M. E. J. (2018). *Networks: An introduction*. New York: Oxford University Press, 2nd edition. doi:10.1093/oso/9780198805090.001.0001.

Niu, Y., Kong, D., Liu, L., Wen, R., & Xiao, J. (2023). Overlapping community detection with adaptive density peaks clustering and iterative partition strategy. *Expert Systems with Applications*, 213, 119213. doi:10.1016/j.eswa.2022.119213.

Ozer, M., Kim, N., & Davulcu, H. (2016). Community detection in political twitter networks using nonnegative matrix factorization methods. In *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)* (pp. 81–88). doi:10.1109/ASONAM.2016.7752217.

Palla, G., Barabási, A.-L., & Vicsek, T. (2007). Quantifying social group evolution. *Nature*, 446(7136), 664–667. doi:10.1038/nature05670.

Palla, G., Derényi, I., Farkas, I., & Vicsek, T. (2005). Uncovering the overlapping community structure of complex networks in nature and society. *nature*, 435, 814–818. doi:10.1038/nature03607.

Pfeiffer III, J. J., Moreno, S., La Fond, T., Neville, J., & Gallagher, B. (2014). Attributed graph models: Modeling network structure with correlated attributes. In *Proceedings of the 23rd international conference on World wide web* (pp. 831–842). doi:10.1145/2566486.2567993.

Pourabbasi, E., Majidnezhad, V., Taghavi Afshord, S., & Jafari, Y. (2021). A new single-chromosome evolutionary algorithm for community detection in complex networks by combining content and structural information. *Expert Systems with Applications*, 186, Article 115854. doi:10.1016/j.eswa.2021.115854.

Psorakis, I., Roberts, S., Ebden, M., & Sheldon, B. (2011). Overlapping community detection using Bayesian non-negative matrix factorization. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 83(6), 1–9. doi:10.1103/PhysRevE.83.066114.

Qin, M., Jin, D., Lei, K., Gabrys, B., & Musial-Gabrys, K. (2018). Adaptive community detection incorporating topology and content in social networks. *Knowledge-Based Systems*, 161, 342–356. doi:10.1016/j.knosys.2018.07.037.

Reihanian, A., Feizi-Derakhshi, M.-R., & Aghdasi, H. S. (2023). An enhanced multi-objective biogeography-based optimization for overlapping community detection in social networks with node attributes. *Information Sciences*, 622, 903–929. doi:10.1016/j.ins.2022.11.125.

Ríos, S. A. & Videla-Cavieres, I. F. (2014). Generating groups of products using graph mining techniques. *Procedia Computer Science*, 35, 730–738. doi:10.1016/j.procs.2014.08.155.

Rossetti, G. & Cazabet, R. (2018). Community Discovery in Dynamic Networks: a Survey. *ACM Computing Surveys*, 51(2), 1–37. doi:10.1145/3172867.

Rossetti, G., Milli, L., & Cazabet, R. (2019). CDLIB: a python library to extract, compare and evaluate communities from complex networks. *Applied Network Science*, 4(1), 1–26. doi:10.1007/s41109-019-0165-9.

Rossetti, G., Pappalardo, L., Pedreschi, D., & Giannotti, F. (2017). Tiles: an online algorithm for community discovery in dynamic social networks. *Machine Learning*, 106(8), 1213–1241. doi:10.1007/s10994-016-5582-8.

Said, A., Abbasi, R. A., Maqbool, O., Daud, A., & Aljohani, N. R. (2018). CC-GA: A clustering coefficient based genetic algorithm for detecting communities in social networks. *Applied Soft Computing*, 63, 59–70. doi:10.1016/j.asoc.2017.11.014.

Shang, R., Zhao, K., Zhang, W., Feng, J., Li, Y., & Jiao, L. (2022). Evolutionary multiobjective overlapping community detection based on similarity matrix and node correction. *Applied Soft Computing*, 127, Article 109397. doi:10.1016/j.asoc.2022.109397.

Sheikholeslami, F. & Giannakis, G. B. (2018). Identification of Overlapping Communities via Constrained Egonet Tensor Decomposition. *IEEE Transactions on Signal Processing*, 66(21), 5730–5745. doi:10.1109/TSP.2018.2871383.

Tan, V. & Févotte, C. (2009). Automatic Relevance Determination in Nonnegative Matrix Factorization. In *SPARS'09 - Signal Processing with Adaptive Sparse Structured Representations* Saint Malo, United Kingdom: HAL. url:inria-00369376.

Tang, F., Wang, C., Su, J., & Wang, Y. (2020). Spectral clustering-based community detection using graph distance and node attributes. *Computational Statistics*, 35(1), 69–94.

doi:10.1007/s00180-019-00909-8.

Taya, F., de Souza, J., Thakor, N. V., & Bezerianos, A. (2016). Comparison method for community detection on brain networks from neuroimaging data. *Applied network science*, 1(1), 1–20. doi:10.1007/s41109-016-0007-y.

Von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and computing*, 17(4), 395–416. doi:10.1007/s11222-007-9033-z.

Wang, W., Jiao, P., He, D., Jin, D., Pan, L., & Gabrys, B. (2016a). Autonomous overlapping community detection in temporal networks: A dynamic Bayesian nonnegative matrix factorization approach. *Knowledge-Based Systems*, 110, 121–134. doi:10.1016/j.knosys.2016.07.021.

Wang, X., Cao, X., Jin, D., Cao, Y., & He, D. (2016b). The (un)supervised NMF methods for discovering overlapping communities as well as hubs and outliers in networks. *Physica A: Statistical Mechanics and its Applications*, 446, 22–34. doi:10.1016/j.physa.2015.11.016.

Wu, H., Gao, L., Dong, J., & Yang, X. (2014). Detecting overlapping protein complexes by rough-fuzzy clustering in protein-protein interaction networks. *PloS one*, 9(3), e91856. doi:10.1371/journal.pone.0091856.

Xie, J., Kelley, S., & Szymanski, B. K. (2013). Overlapping Community Detection in Networks: The State-of-the-art and Comparative Study. *ACM Comput. Surv.*, 45(4), 43:1–43:35. doi:10.1145/2501654.2501657.

Yang, J., McAuley, J., & Leskovec, J. (2013). Community detection in networks with node attributes. In *2013 IEEE 13th International Conference on Data Mining (ICDM)* (pp. 1151–1156). doi:10.1109/ICDM.2013.167.

Yu, W., Wang, W., Jiao, P., & Li, X. (2019). Evolutionary clustering via graph regularized nonnegative matrix factorization for exploring temporal networks. *Knowledge-Based Systems*, 167, 1–10. doi:10.1016/j.knosys.2019.01.024.

Zhou, Y., Cheng, H., & Yu, J. X. (2009). Graph clustering based on structural/attribute similarities. *Proceedings of the VLDB Endowment*, 2(1), 718–729. doi:10.14778/1687627.1687709.

Annexes

Annex A

Detailed computations of the model for timestep 1

We will generalize each parameter computation at timestamp t, since they will appear at all time steps. The first term is the negative log likelihood of the data $\mathbf{V}_t$. Since its expectation is based on $\mathbf{W}_t$ and $\mathbf{H}_t$, this can be rewritten has $\mathbb{P}\left(\mathbf{V}_t|\mathbf{W}_t,\mathbf{H}_t\right) = \mathbb{P}\left(\mathbf{V}_t|\hat{\mathbf{V}}_t\right)$, representing the probability of observing every interaction $v_{\mathrm{i}j,t}$ given a Poisson rate $\hat{v}_{\mathrm{i}j,t}$ Psorakis et al. (2011). Accordingly, the negative log likelihood of observation $v_{\mathrm{i}j,t}$ is represented as stated in Eq. (A.1).

$$-\log\mathbb{P}\left(v_{\mathrm{i}j,t}|\hat{v}_{\mathrm{i}j,t}\right) = -\log\left(\frac{\exp\left(-\hat{v}_{\mathrm{i}j,t}\right)\hat{v}_{\mathrm{i}j,t}^{v_{\mathrm{i}j,t}}}{v_{\mathrm{i}j,t}!}\right) = \hat{v}_{\mathrm{i}j,t} - v_{\mathrm{i}j,t}\log\hat{v}_{\mathrm{i}j,t} + \log v_{\mathrm{i}j,t}! \tag{A.1}$$

Last term of Eq. (A.1) can be approximated by using the Stirling approximation to second order, as stated in Eq. (A.2).

$$\log v_{\mathrm{i}j,t}! \approx \log\left(\sqrt{2\pi v_{\mathrm{i}j,t}}\left(\frac{v_{\mathrm{i}j,t}}{\mathrm{e}}\right)^{v_{\mathrm{i}j,t}}\right) = \frac{1}{2}\log\left(2\pi v_{\mathrm{i}j,t}\right) + v_{\mathrm{i}j,t}\log v_{\mathrm{i}j,t} - v_{\mathrm{i}j,t} \tag{A.2}$$

Replacing Eq. (A.2) in Eq. (A.1), we obtain a new expression for negative log likelihood of observation $v_{\mathrm{i}j,t}$, that can be rewritten as Eq. (A.3).

$$-\log\mathbb{P}\left(v_{\mathrm{i}j,t}|\hat{v}_{\mathrm{i}j,t}\right) \approx v_{\mathrm{i}j,t}\log\left(\frac{v_{\mathrm{i}j,t}}{\hat{v}_{\mathrm{i}j,t}}\right) + \hat{v}_{\mathrm{i}j,t} - v_{\mathrm{i}j,t} + \frac{1}{2}\log\left(2\pi v_{\mathrm{i}j,t}\right) \tag{A.3}$$

Since the loss function includes all observations, a full negative log likelihood for the entire adjacency matrix is formulated in Eq. (A.4), considering independence between each link.

$$-\log \mathbb{P}\left(\mathbf{V}_t|\hat{\mathbf{V}}_t\right) = -\log \prod_{\mathrm{i},j} \mathbb{P}\left(v_{\mathrm{i}j,t}|\hat{v}_{\mathrm{i}j,t}\right) = -\sum_{\mathrm{i}=1}^{N_t}\sum_{j=1}^{N_t} \log \mathbb{P}\left(v_{\mathrm{i}j,t}|\hat{v}_{\mathrm{i}j,t}\right) \simeq$$
$$\sum_{\mathrm{i}=1}^{N_t}\sum_{j=1}^{N_t}\left(v_{\mathrm{i}j,t}\log\left(\frac{v_{\mathrm{i}j,t}}{\hat{v}_{\mathrm{i}j,t}}\right) + \hat{v}_{\mathrm{i}j,t} - v_{\mathrm{i}j,t} + \frac{1}{2}\log\left(2\pi v_{\mathrm{i}j,t}\right)\right) \tag{A.4}$$

Analogously, second term of can be written as Eq. (A.5), considering independence between the generation of each link, and between the generation of each attribute.

$$-\log \mathbb{P}\left(\mathbf{F}_t|\hat{\mathbf{F}}_t\right) = -\log \prod_{m,j} \mathbb{P}\left(f_{mj,t}|\hat{f}_{mj,t}\right) = -\sum_{m=1}^{M}\sum_{j=1}^{N_t} \log \mathbb{P}\left(f_{mj,t}|\hat{f}_{mj,t}\right) \simeq$$
$$\sum_{m=1}^{M}\sum_{j=1}^{N_t}\left(f_{mj,t}\log\left(\frac{f_{mj,t}}{\hat{f}_{mj,t}}\right) + \hat{f}_{mj,t} - f_{mj,t} + \frac{1}{2}\log\left(2\pi f_{mj,t}\right)\right) \tag{A.5}$$

Since $\mathbf{W}_t$, $\mathbf{H}_t$ and $\mathbf{H}_t$ are non-negative matrices, probability distributions to model them need to fulfill this property, such as a Half Normal, Exponential or Gamma distribution. In this case, the third, fourth and fifth terms of Eq. (2.3) are determined considering half normal priors over the columns of $\mathbf{W}_t$, rows of $\mathbf{H}_t$, and columns of $\mathbf{G}_t$, with scale parameters $\boldsymbol{\beta}_t \in \mathbb{R}^K$.

Half normal distribution and its application into each $w_{\mathrm{i}k,t}$ term with scale parameter $\beta_{k,t}$ ($\mathcal{HN}(0, \beta_{k,t}^{-1})$) are shown in Eq. (A.6), considering $\beta_{k,t}^{-1} = \sigma^2$.

$$\mathbb{P}\left(x|\,\sigma\right) = \frac{\sqrt{2}}{\sigma\sqrt{\pi}}\exp\left(\frac{-x^2}{2\sigma^2}\right), \quad x > 0 \Rightarrow$$
$$\mathbb{P}\left(w_{\mathrm{i}j,t}|\,\beta_{k,t}\right) = \frac{\sqrt{2}}{\beta_{k,t}^{-1/2}\sqrt{\pi}}\exp\left(\frac{-w_{\mathrm{i}k,t}^2}{2\beta_{k,t}^{-1}}\right), \quad \beta_{k,t} > 0 \tag{A.6}$$

Now, considering all observations, independence between each node assignment into a community k and taking the logarithm, the third term of Eq. (2.3) is defined as Eq. (A.7).

$$\mathbb{P}\left(\mathbf{W}_t|\boldsymbol{\beta}_t\right) = \prod_{\mathrm{i},k} \frac{\sqrt{2}}{\sqrt{\pi}} \sqrt{\beta_{k,t}} \exp\left(-\frac{w_{\mathrm{i}k,t}^2 \beta_{k,t}}{2}\right) \Rightarrow$$

$$-\log \mathbb{P}\left(\mathbf{W}_t|\boldsymbol{\beta}_t\right) = -\sum_{\mathrm{i}=1}^{N_t} \sum_{k=1}^{K} \log\left(\frac{\sqrt{2}}{\sqrt{\pi}} \sqrt{\beta_{k,t}} \exp\left(-\frac{w_{\mathrm{i}k,t}^2 \beta_{k,t}}{2}\right)\right)$$

$$= \sum_{\mathrm{i}=1}^{N_t} \sum_{k=1}^{K} \left(\frac{w_{\mathrm{i}k,t}^2 \beta_{k,t}}{2} - \log\left(\beta_{k,t}^{1/2}\right) - \log\frac{2^{1/2}}{\pi^{1/2}}\right) =$$

$$\sum_{\mathrm{i}=1}^{N_t} \sum_{k=1}^{K} \frac{w_{\mathrm{i}k,t}^2 \beta_{k,t}}{2} - \frac{N_t}{2} \sum_{k=1}^{K} \log \beta_{k,t} - N_t K \log\frac{\sqrt{2}}{\sqrt{\pi}} \tag{A.7}$$

Similarly, third and fourth term of Eq. (2.3) are defined as Eqs. (A.8) and (A.9).

$$-\log \mathbb{P}\left(\mathbf{H}_t|\boldsymbol{\beta}_t\right) = \sum_{k=1}^{K} \sum_{j=1}^{N_t} \frac{h_{kj,t}^2 \beta_{k,t}}{2} - \frac{N_t}{2} \sum_{k=1}^{K} \log \beta_{k,t} - N_t K \log\frac{\sqrt{2}}{\sqrt{\pi}} \tag{A.8}$$

$$-\log \mathbb{P}\left(\mathbf{G}_t|\boldsymbol{\beta}_t\right) = \sum_{m=1}^{M} \sum_{k=1}^{K} \frac{g_{mk,t}^2 \beta_{k,t}}{2} - \frac{M}{2} \sum_{k=1}^{K} \log \beta_{k,t} - MK \log\frac{\sqrt{2}}{\sqrt{\pi}} \tag{A.9}$$

Finally, each $\beta_{k,t}$ also has to be defined with a non-negative distribution, in this case, a Gamma distribution was defined with fixed hyperparameters $a_{k,t}$, $b_{k,t}$ Psorakis et al. (2011). Both parameters are considered constant for each community and each timestamp. Eq. (A.10) shows the calculation of the last term in Eq. (2.3), considering independence between each $\beta_{k,t}$.

$$\mathbb{P}\left(\beta_{k,t}|a_{k,t}, b_{k,t}\right) = \frac{\beta_{k,t}^{a_{k,t}-1} \exp\left(-\beta_{k,t} b_{k,t}\right) b_{k,t}^{a_{k,t}}}{\Gamma\left(a_{k,t}\right)} \Rightarrow$$

$$\mathbb{P}\left(\boldsymbol{\beta}_t|\mathbf{a}_t, \mathbf{b}_t\right) = \prod_{k=1}^{K} \frac{\beta_{k,t}^{a_{k,t}-1} \exp\left(-\beta_{k,t} b_{k,t}\right) b_{k,t}^{a_{k,t}}}{\Gamma\left(a_k\right)} \Rightarrow -\log \mathbb{P}\left(\boldsymbol{\beta}_t|\mathbf{a}_t, \mathbf{b}_t\right) =$$

$$\sum_{k=1}^{K} \left(\beta_k b_{k,t} - \left(a_{k,t} - 1\right) \log \beta_{k,t}\right) + K\left(\log \Gamma\left(a_{k,t}\right) - a_{k,t} \log b_{k,t}\right) \tag{A.10}$$

Annex B

Detailed computations of the model for timestep t

Eqs. (B.1) and (B.2) show the calculations of the third term of Eq. (2.7). The fourth term of Eq. (2.7) is obtained by equation (B.3) and the fifth term is obtained by equation (B.4). In all cases $t = 2, \ldots, T$.

$$\mathbb{P}\left(w_{\mathrm{i}k,t}|\gamma_{\mathrm{i}k,t},\mu,\beta_{k,t}\right) = \frac{w_{\mathrm{i}k,t}^{\gamma_{\mathrm{i}k,t}-1}\exp\left(-w_{\mathrm{i}k,t}\mu\right)\mu^{\gamma_{\mathrm{i}k,t}}}{\Gamma\left(\gamma_{\mathrm{i}k,t}\right)}\frac{\sqrt{2}}{\beta_{k,t}^{-1/2}\sqrt{\pi}}\exp\left(\frac{-w_{\mathrm{i}k,t}^2}{2\beta_{k,t}^{-1}}\right) \Rightarrow$$

$$\mathbb{P}\left(\mathbf{W}_t|\boldsymbol{\gamma}_t,\mu,\boldsymbol{\beta}_t\right) = \prod_{\mathrm{i}=1}^{N_t}\prod_{k=1}^{K}\frac{w_{\mathrm{i}k,t}^{\gamma_{\mathrm{i}k,t}-1}\exp\left(-w_{\mathrm{i}k,t}\mu\right)\mu^{\gamma_{\mathrm{i}k,t}}}{\Gamma\left(\gamma_{\mathrm{i}k,t}\right)}\frac{\sqrt{2}}{\beta_{k,t}^{-1/2}\sqrt{\pi}}\exp\left(\frac{-w_{\mathrm{i}k,t}^2}{2\beta_{k,t}^{-1}}\right) \Rightarrow$$

$$-\log\mathbb{P}\left(\mathbf{W}_t|\boldsymbol{\gamma}_t,\mu,\boldsymbol{\beta}_t\right) = \sum_{\mathrm{i}=1}^{N_t}\sum_{k=1}^{K}\Bigg[-\left(\gamma_{\mathrm{i}k,t}-1\right)\log w_{\mathrm{i}k,t} + w_{\mathrm{i}k,t}\mu$$

$$-\gamma_{\mathrm{i}k,t}\log\mu + \log\Gamma\left(\gamma_{\mathrm{i}k,t}\right) + \frac{w_{\mathrm{i}k,t}^2\beta_{k,t}}{2} - \log\left(\beta_{k,t}^{1/2}\right) - \log\frac{2^{1/2}}{\pi^{1/2}}\Bigg] \qquad \text{(B.1)}$$

Since we defined $\gamma_{\mathrm{i}k,t} = \mu w^{''}_{\mathrm{i}k,t-1} + 1$, equation (B.1) transforms into equation (B.2).

$$
\begin{aligned}
-\log \mathbb{P}\left(\mathbf{W}_t|\mathbf{W}^{''}_{t-1}, \mu, \boldsymbol{\beta}_t\right) = \sum_{\mathrm{i}=1}^{N_t}\sum_{k=1}^{K}\Bigg[& -\left(\mu w^{''}_{\mathrm{i}k,t-1} + 1 - 1\right)\log w_{\mathrm{i}k,t} + w_{\mathrm{i}k,t}\mu \\
& -\left(\mu w^{''}_{\mathrm{i}k,t-1} + 1\right)\log\mu + \log\Gamma\left(\mu w^{''}_{\mathrm{i}k,t-1} + 1\right) + \frac{w^2_{\mathrm{i}k,t}\beta_{k,t}}{2} - \log\left(\beta^{1/2}_{k,t}\right) \\
& -\log\frac{2^{1/2}}{\pi^{1/2}}\Bigg] = \sum_{\mathrm{i}=1}^{N_t}\sum_{k=1}^{K}\Bigg[\mu\left(-w^{''}_{\mathrm{i}k,t-1}\log w_{\mathrm{i}k,t} + w_{\mathrm{i}k,t}\right) + \frac{w^2_{\mathrm{i}k,t}\beta_{k,t}}{2} - \frac{1}{2}\log\beta_{k,t} + \\
& \kappa\Bigg] = \sum_{\mathrm{i}=1}^{N_t}\sum_{k=1}^{K}\mu\left(-w^{''}_{\mathrm{i}k,t-1}\log w_{\mathrm{i}k,t} + w_{\mathrm{i}k,t}\right) + \sum_{\mathrm{i}=1}^{N_t}\sum_{k=1}^{K}\frac{w^2_{\mathrm{i}k,t}\beta_{k,t}}{2} \\
& -\frac{N_t}{2}\sum_{k=1}^{K}\log\beta_{k,t} + \kappa
\end{aligned}
\tag{B.2}
$$

$$
\begin{aligned}
& -\log\mathbb{P}\left(\mathbf{H}_t|\mathbf{H}^{''}_{t-1}, \mu, \boldsymbol{\beta}_t\right) = \\
& \sum_{j=1}^{N_t}\sum_{k=1}^{K}\mu\left(-h^{''}_{kj,t-1}\log h_{kj,t} + h_{kj,t}\right) + \sum_{j=1}^{N_t}\sum_{k=1}^{K}\frac{h^2_{kj,t}\beta_{k,t}}{2} - \frac{N_t}{2}\sum_{k=1}^{K}\log\beta_{k,t} + \kappa
\end{aligned}
\tag{B.3}
$$

$$
\begin{aligned}
& -\log\mathbb{P}\left(\mathbf{G}_t|\mathbf{G}^{''}_{t-1}, \mu, \boldsymbol{\beta}_t\right) = \\
& \sum_{\mathrm{i}=1}^{M}\sum_{k=1}^{K}\mu\left(-g_{mk,t-1}\log g_{mk,t} + g_{mk,t}\right) + \sum_{\mathrm{i}=1}^{M}\sum_{k=1}^{K}\frac{g^2_{mk,t}\beta_{k,t}}{2} - \frac{M}{2}\sum_{k=1}^{K}\log\beta_{k,t} + \kappa
\end{aligned}
\tag{B.4}
$$

Annex C

Partial derivatives of the loss function with respect to the latent factors at time step 1

Eqs. (C.1)-(C.4), show the scalar form of the partial derivatives of the loss function from (2.4), with respect to $\mathbf{H}_1$, $\mathbf{W}_1$, $\mathbf{G}_1$, and $\beta_{k,1}$, respectively.

$$\begin{aligned}\frac{\partial \mathcal{U}_1}{\partial h_{k'j',1}} &= \frac{\partial}{\partial h_{k'j',1}} \sum_{\mathrm{i}=1}^{N_1}\sum_{j=1}^{N_1}\sum_{k=1}^{K} w_{\mathrm{i}k,1}h_{kj,1} - \frac{\partial}{\partial h_{k'j',1}} \sum_{\mathrm{i}=1}^{N_1}\sum_{j=1}^{N_1} v_{\mathrm{i}j,1}\log\sum_{k=1}^{K} w_{\mathrm{i}k,1}h_{kj,1} \\ &+\frac{\partial}{\partial h_{k'j',1}} \sum_{m=1}^{M}\sum_{j=1}^{N_1}\sum_{k=1}^{K} g_{mk,1}h_{kj,1} - \frac{\partial}{\partial h_{k'j',1}} \sum_{m=1}^{M}\sum_{j=1}^{N_1} f_{mj,1}\log\sum_{k=1}^{K} g_{mk,1}h_{kj,1} \\ &+\frac{1}{2}\frac{\partial}{\partial h_{k'j',1}} \sum_{k=1}^{K}\sum_{j=1}^{N_1}\beta_{k,1}h_{kj,1}^2 = \sum_{\mathrm{i}=1}^{N_1} w_{\mathrm{i}k',1} - \sum_{\mathrm{i}=1}^{N_1}\frac{v_{\mathrm{i}j',1}w_{\mathrm{i}k',1}}{\sum_{k=1}^{K} w_{\mathrm{i}k,1}h_{kj',1}} \\ &+\sum_{m=1}^{M} g_{mk',1} - \sum_{m=1}^{M}\frac{f_{mj',1}g_{mk',1}}{\sum_{k=1}^{K} g_{mk,1}h_{kj',1}} + \beta_{k',1}h_{k'j',1} \end{aligned} \tag{C.1}$$

$$\begin{aligned}\frac{\partial \mathcal{U}_1}{\partial w_{\mathrm{i}'k',1}} &= \frac{\partial}{\partial w_{\mathrm{i}'k',1}} \sum_{\mathrm{i}=1}^{N_1}\sum_{j=1}^{N_1}\sum_{k=1}^{K} w_{\mathrm{i}k,1}h_{kj,1} - \frac{\partial}{\partial w_{\mathrm{i}'k',1}} \sum_{\mathrm{i}=1}^{N_1}\sum_{j=1}^{N_1} v_{\mathrm{i}j,1}\log\sum_{k=1}^{K} w_{\mathrm{i}k,1}h_{kj,1} \\ &+\frac{1}{2}\frac{\partial}{\partial w_{\mathrm{i}'k',1}} \sum_{k=1}^{K}\sum_{\mathrm{i}=1}^{N_1}\beta_{k,1}\left(w_{\mathrm{i}k,1}^2\right) = \sum_{j=1}^{N_1} h_{k'j,1} - \sum_{j=1}^{N_1}\frac{v_{\mathrm{i}'j,1}h_{k'j,1}}{\sum_{k=1}^{K} w_{\mathrm{i}'k,1}h_{kj,1}} + \beta_{k',1}w_{\mathrm{i}'k',1} \end{aligned} \tag{C.2}$$

$$\begin{aligned}\frac{\partial \mathcal{U}_1}{\partial g_{m'k',1}} &= \frac{\partial}{\partial g_{m'k',1}} \sum_{m=1}^{M}\sum_{j=1}^{N_1}\sum_{k=1}^{K} g_{mk,1}h_{kj,1} - \frac{\partial}{\partial g_{m'k',1}} \sum_{m=1}^{M}\sum_{j=1}^{N_1} f_{mj,1}\log\sum_{k=1}^{K} g_{mk,1}h_{kj,1} \\ &+\frac{1}{2}\frac{\partial}{\partial g_{m'k',1}} \sum_{k=1}^{K}\sum_{m=1}^{M}\beta_{k,1}\left(g_{mk,1}^2\right) = \sum_{j=1}^{N_1} h_{k'j,1} - \sum_{j=1}^{N_1}\frac{f_{m'j,1}h_{k'j,1}}{\sum_{k=1}^{K} g_{m'k,1}h_{kj,1}} + \beta_{k',1}g_{m'k',1} \end{aligned} \tag{C.3}$$

$$\frac{\partial \mathcal{U}_1}{\partial \beta_{k,1}} = \frac{1}{2}\sum_{\mathrm{i}=1}^{N_1} w_{\mathrm{ik},1}^2 + \frac{1}{2}\sum_{j=1}^{N_1} h_{kj,1}^2 + \frac{1}{2}\sum_{m=1}^{M} g_{mk,1}^2 - \frac{N_1 + M/2}{\beta_{k,1}} + b_{k,1} - \frac{a_{k,1} - 1}{\beta_{k,1}} \tag{C.4}$$

Eqs. (C.5)-(C.7) show the matrix form of the partial derivatives of the loss function from (2.4), with respect to the latent factors, where $\mathbf{B}_1 \in \mathbb{R}^{K\times K}$ is a matrix with elements $\beta_{k,1}$ in the diagonal and zero elsewhere.

$$\begin{aligned}\nabla_{H_1}\mathcal{U}_1 = (\mathbf{W}_{\mathbf{1}N_1\times K})^T \left(\frac{\mathbf{W}_{\mathbf{1}N_1\times K}\mathbf{H}_{\mathbf{1}K\times N_1} - \mathbf{V}_{\mathbf{1}N_1\times N_1}}{\mathbf{W}_{\mathbf{1}N_1\times K}\mathbf{H}_{\mathbf{1}K\times N_1}}\right) \\ + (\mathbf{G}_{\mathbf{1}M\times K})^T \left(\frac{\mathbf{G}_{\mathbf{1}M\times K}\mathbf{H}_{\mathbf{1}K\times N_1} - \mathbf{F}_{\mathbf{1}M\times N_1}}{\mathbf{G}_{\mathbf{1}M\times K}\mathbf{H}_{\mathbf{1}K\times N_1}}\right) + \mathbf{B}_{\mathbf{1}K\times K}\mathbf{H}_{\mathbf{1}K\times N_1}\end{aligned} \tag{C.5}$$

$$\nabla_{W_1}\mathcal{U}_1 = \left(\frac{\mathbf{W}_{\mathbf{1}N_1\times K}\mathbf{H}_{\mathbf{1}K\times N_1} - \mathbf{V}_{\mathbf{1}N_1\times N_1}}{\mathbf{W}_{\mathbf{1}N_1\times K}\mathbf{H}_{\mathbf{1}K\times N_1}}\right)(\mathbf{H}_{\mathbf{1}K\times N_1})^T + \mathbf{W}_{\mathbf{1}N_1\times K}\mathbf{B}_{\mathbf{1}K\times K} \tag{C.6}$$

$$\nabla_{G_1}\mathcal{U}_1 = \left(\frac{\mathbf{G}_{\mathbf{1}M\times K}\mathbf{H}_{\mathbf{1}K\times N_1} - \mathbf{F}_{\mathbf{1}M\times N_1}}{\mathbf{G}_{\mathbf{1}M\times K}\mathbf{H}_{\mathbf{1}K\times N_1}}\right)(\mathbf{H}_{\mathbf{1}K\times N_1})^T + \mathbf{G}_{\mathbf{1}M\times K}\mathbf{B}_{\mathbf{1}K\times K} \tag{C.7}$$

Annex D

Partial derivatives of the loss function with respect to the latent factors at time step t

Eqs. (D.1)-(D.4), show the scalar form of the partial derivatives of the loss function from (2.8), with respect to $\mathbf{H}_t$, $\mathbf{W}_t$, $\mathbf{G}_t$, and $\beta_{k,1}$, respectively.

$$\begin{aligned}\frac{\partial \mathcal{U}_t}{\partial h_{k'j',t}} &= \frac{\partial}{\partial h_{k'j',t}} \sum_{\mathrm{i}=1}^{N_1}\sum_{j=1}^{N_t}\sum_{k=1}^{K} w_{\mathrm{i}k,t}h_{kj,t} - \frac{\partial}{\partial h_{k'j',t}} \sum_{\mathrm{i}=1}^{N_t}\sum_{j=1}^{N_t} v_{\mathrm{i}j,t}\log\sum_{k=1}^{K} w_{\mathrm{i}k,t}h_{kj,t} \\ &+\frac{\partial}{\partial h_{k'j',t}} \sum_{m=1}^{M}\sum_{j=1}^{N_t}\sum_{k=1}^{K} g_{mk,t}h_{kj,t} - \frac{\partial}{\partial h_{k'j',t}} \sum_{m=1}^{M}\sum_{j=1}^{N_t} f_{mj,t}\log\sum_{k=1}^{K} g_{mk,t}h_{kj,t} \\ &-\frac{\partial}{\partial h_{k'j',t}} \sum_{j=1}^{N_t}\sum_{k=1}^{K} \mu h''_{kj,t-1}\log h_{kj,t} + \frac{\partial}{\partial h_{k'j',t}} \sum_{j=1}^{N_t}\sum_{k=1}^{K} \mu h_{kj,t} \\ &+\frac{1}{2}\frac{\partial}{\partial h_{k'j',t}} \sum_{k=1}^{K}\sum_{j=1}^{N_t} \beta_{k,t}h_{kj,t}^2 = \sum_{\mathrm{i}=1}^{N_t} w_{\mathrm{i}k',t} - \sum_{\mathrm{i}=1}^{N_t} \frac{v_{\mathrm{i}j',t}w_{\mathrm{i}k',t}}{\sum_{k=1}^{K} w_{\mathrm{i}k,t}h_{kj',t}} + \sum_{m=1}^{M} g_{mk',t} \\ &-\sum_{m=1}^{M} \frac{f_{mj',t}g_{mk',t}}{\sum_{k=1}^{K} g_{mk,t}h_{kj',t}} - \mu\frac{h''_{k'j',t-1}}{h_{k'j',t}} + \mu + \beta_{k',t}h_{k'j',t} \end{aligned} \tag{D.1}$$

$$\begin{aligned}\frac{\partial \mathcal{U}_t}{\partial w_{\mathrm{i}'k',t}} &= \frac{\partial}{\partial w_{\mathrm{i}'k',t}} \sum_{\mathrm{i}=1}^{N_t}\sum_{j=1}^{N_t}\sum_{k=1}^{K} w_{\mathrm{i}k,t}h_{kj,t} - \frac{\partial}{\partial w_{\mathrm{i}'k',t}} \sum_{\mathrm{i}=1}^{N_t}\sum_{j=1}^{N_t} v_{\mathrm{i}j,t}\log\sum_{k=1}^{K} w_{\mathrm{i}k,t}h_{kj,t} \\ &-\frac{\partial}{\partial w_{\mathrm{i}'k',t}} \sum_{\mathrm{i}=1}^{N_t}\sum_{k=1}^{K} \mu w''_{\mathrm{i}k,t-1}\log w_{\mathrm{i}k,t} + \frac{\partial}{\partial w_{\mathrm{i}'k',t}} \sum_{\mathrm{i}=1}^{N_t}\sum_{k=1}^{K} \mu w_{\mathrm{i}k,t} + \frac{1}{2}\frac{\partial}{\partial w_{\mathrm{i}'k',t}} \sum_{k=1}^{K}\sum_{\mathrm{i}=1}^{N_t} \beta_{k,t}w_{\mathrm{i}k,t}^2 = \\ &\sum_{j=1}^{N_t} h_{k'j,t} - \sum_{j=1}^{N_t} \frac{v_{\mathrm{i}'j,t}h_{k'j,t}}{\sum_{k=1}^{K} w_{\mathrm{i}'k,t}h_{kj,t}} - \mu\frac{w''_{\mathrm{i}'k',t-1}}{w_{\mathrm{i}'k',t}} + \mu + \beta_{k',t}w_{\mathrm{i}'k',t} \end{aligned} \tag{D.2}$$

$$\frac{\partial \mathcal{U}_t}{\partial g_{m'k',t}} = \frac{\partial}{\partial g_{m'k',t}} \sum_{m=1}^{M}\sum_{j=1}^{N_t}\sum_{k=1}^{K} g_{mk,t}h_{kj,t} - \frac{\partial}{\partial g_{m'k',t}} \sum_{m=1}^{M}\sum_{j=1}^{N_t} f_{mj,t} \log \sum_{k=1}^{K} g_{mk,t}h_{kj,t}$$
$$-\frac{\partial}{\partial g_{m'k',t}} \sum_{m=1}^{M}\sum_{k=1}^{K} \mu g_{mk,t-1} \log g_{mk,t} + \frac{\partial}{\partial g_{m'k',t}} \sum_{m=1}^{M}\sum_{k=1}^{K} \mu g_{mk,t} + \frac{1}{2}\frac{\partial}{\partial g_{m'k',t}} \sum_{k=1}^{K}\sum_{m=1}^{M} \beta_{k,t} g_{mk,t}^2$$
$$= \sum_{j=1}^{N_t} h_{k'j,t} - \sum_{j=1}^{N_t} \frac{f_{m'j,t}h_{k'j,t}}{\sum_{k=1}^{K} g_{m'k,t}h_{kj,t}} - \mu \frac{g_{m'k',t-1}}{g_{m'k',t}} + \mu + \beta_{k',t} g_{m'k',t} \quad \text{(D.3)}$$

$$\frac{\partial \mathcal{U}_t}{\partial \beta_{k,t}} = \frac{1}{2}\sum_{\mathrm{i}=1}^{N_t} w_{\mathrm{ik},t}^2 + \frac{1}{2}\sum_{j=1}^{N_t} h_{kj,t}^2 + \frac{1}{2}\sum_{m=1}^{M} g_{mk,t}^2 - \frac{N_t + M/2}{\beta_{k,t}} + b_{k,t} - \frac{a_{k,t} - 1}{\beta_{k,t}} \quad \text{(D.4)}$$

Eqs. (D.5)-(D.7) show the matrix form of the partial derivatives of the loss function from (2.8), with respect to the latent factors, where $\mathbf{B}_t \in \mathbb{R}^{K\times K}$ is a matrix with elements $\beta_{k,t}$ in the diagonal and zero elsewhere.

$$\nabla_{H_t}\mathcal{U}_t = (\mathbf{W_t}_{N_t\times K})^T \left(\frac{\mathbf{W_t}_{N_t\times K}\mathbf{H_t}_{K\times N_t} - \mathbf{V_t}_{N_t\times N_t}}{\mathbf{W_t}_{N_t\times K}\mathbf{H_t}_{K\times N_t}}\right)$$
$$+ (\mathbf{G_t}_{M\times K})^T \left(\frac{\mathbf{G_t}_{M\times K}\mathbf{H_t}_{K\times N_t} - \mathbf{F_t}_{M\times N_t}}{\mathbf{G_t}_{M\times K}\mathbf{H_t}_{K\times N_t}}\right)$$
$$-\mu\left(\frac{\mathbf{H''_{t-1}}_{K\times N_t}}{\mathbf{H_t}_{K\times N_t}} - \mathbf{1}_{K\times N_t}\right) + \mathbf{B_t}_{K\times K}\mathbf{H_t}_{K\times N_t} \quad \text{(D.5)}$$

$$\nabla_{W_t}\mathcal{U}_t = \left(\frac{\mathbf{W_t}_{N_t\times K}\mathbf{H_t}_{K\times N_t} - \mathbf{V_t}_{N_t\times N_t}}{\mathbf{W_t}_{N_t\times K}\mathbf{H_t}_{K\times N_t}}\right)(\mathbf{H_t}_{K\times N_t})^T$$
$$-\mu\left(\frac{\mathbf{W''_{t-1}}_{N_t\times K}}{\mathbf{W_t}_{N_t\times K}} - \mathbf{1}_{N_t\times K}\right) + \mathbf{W_t}_{N_t\times K}\mathbf{B_t}_{K\times K} \quad \text{(D.6)}$$

$$\nabla_{G_t}\mathcal{U}_t = \left(\frac{\mathbf{G_t}_{M\times K}\mathbf{H_t}_{K\times N_t} - \mathbf{F_t}_{M\times N_t}}{\mathbf{G_t}_{M\times K}\mathbf{H_t}_{K\times N_t}}\right)(\mathbf{H_t}_{K\times N_t})^T$$
$$-\mu\left(\frac{\mathbf{G_{t-1}}_{M\times K}}{\mathbf{G_t}_{M\times K}} - \mathbf{1}_{M\times K}\right) + \mathbf{G_t}_{M\times K}\mathbf{B_t}_{K\times K} \quad \text{(D.7)}$$

Annex E

Multiplicative coordinate descent algorithm equations

Eqs. (E.1)-(E.3), show the results from applying the multiplicative coordinate descent algorithm to Eqs. (D.5)-(D.7), respectively.

$$\mathbf{H}_{\mathbf{t}K\times N_t} \leftarrow \left(\frac{\mathbf{H}_{\mathbf{t}K\times N_t}}{\left(\mathbf{W}_{\mathbf{t}N_t\times K}\right)^T \mathbf{1}_{N_t\times N_t} + \left(\mathbf{G}_{\mathbf{t}M\times K}\right)^T \mathbf{1}_{M\times N_t} + \frac{1-\alpha}{\alpha}\mathbf{1}_{K\times N_t} + \mathbf{B}_{\mathbf{t}K\times K}\mathbf{H}_{\mathbf{t}K\times N_t}}\right) \circ \left[\left(\mathbf{W}_{\mathbf{t}N_t\times K}\right)^T \frac{\mathbf{V}_{\mathbf{t}N_t\times N_t}}{\mathbf{W}_{\mathbf{t}N_t\times K}\mathbf{H}_{\mathbf{t}K\times N_t}} + \left(\mathbf{G}_{\mathbf{t}M\times K}\right)^T \frac{\mathbf{F}_{\mathbf{t}M\times N_t}}{\mathbf{G}_{\mathbf{t}M\times K}\mathbf{H}_{\mathbf{t}K\times N_t}} + \frac{1-\alpha}{\alpha}\frac{\mathbf{H}''_{\mathbf{t-1}K\times N_t}}{\mathbf{H}_{\mathbf{t}K\times N_t}}\right] \tag{E.1}$$

$$\mathbf{W}_{\mathbf{t}N_t\times K} \leftarrow \left(\frac{\mathbf{W}_{\mathbf{t}N_t\times K}}{\mathbf{1}_{N_t\times N_t}\left(\mathbf{H}_{\mathbf{t}K\times N_t}\right)^T + \frac{1-\alpha}{\alpha}\mathbf{1}_{N_t\times K} + \mathbf{W}_{\mathbf{t}N_t\times K}\mathbf{B}_{\mathbf{t}K\times K}}\right) \circ \left[\left(\frac{\mathbf{V}_{\mathbf{t}N_t\times N_t}}{\mathbf{W}_{\mathbf{t}N_t\times K}\mathbf{H}_{\mathbf{t}K\times N_t}}\left(\mathbf{H}_{\mathbf{t}K\times N_t}\right)^T\right) + \frac{1-\alpha}{\alpha}\frac{\mathbf{W}''_{\mathbf{t-1}N_t\times K}}{\mathbf{W}_{\mathbf{t}N_t\times K}}\right] \tag{E.2}$$

$$\mathbf{G}_{\mathbf{t}M\times K} \leftarrow \left(\frac{\mathbf{G}_{\mathbf{t}M\times K}}{\mathbf{1}_{M\times N_t}\left(\mathbf{H}_{\mathbf{t}K\times N_t}\right)^T + \frac{1-\alpha}{\alpha}\mathbf{1}_{M\times K} + \mathbf{G}_{\mathbf{t}M\times K}\mathbf{B}_{\mathbf{t}K\times K}}\right) \circ \left[\left(\frac{\mathbf{F}_{\mathbf{t}M\times N_t}}{\mathbf{G}_{\mathbf{t}M\times K}\mathbf{H}_{\mathbf{t}K\times N_t}}\left(\mathbf{H}_{\mathbf{t}K\times N_t}\right)^T\right) + \frac{1-\alpha}{\alpha}\frac{\mathbf{G}_{\mathbf{t-1}M\times K}}{\mathbf{G}_{\mathbf{t}M\times K}}\right] \tag{E.3}$$