Unsupervised approaches for time-evolving graph embeddings with application to human microbiome

Dissertation
zur Erlangung des Grades eines
Doktors der Naturwissenschaften (Dr. rer. nat.)

am Fachbereich Mathematik und Informatik
der Freien Universität Berlin

vorgelegt von

Kateryna Melnyk

Berlin, 2024

Erstgutachter: Prof. Dr. Tim O. F. Conrad
Zweitgutachter: Prof. Dr. Cristofer Englund

Tag der Disputation: February 13, 2024

Abstract

More and more diseases have been found to be strongly correlated with disturbances in the microbiome constitution, e.g., obesity, diabetes, and even some types of cancer. Advances in high-throughput omics technologies have made it possible to directly analyze the human microbiome and its impact on human health and physiology. Microbial composition is usually observed over long periods of time and the interactions between their members are explored. Numerous studies have used microbiome data to accurately differentiate disease states and understand the differences in microbiome profiles between healthy and ill individuals. However, most of them mainly focus on various statistical approaches, omitting microbe-microbe interactions among a large number of microbiome species that, in principle, drive microbiome dynamics. Constructing and analyzing time-evolving graphs is needed to understand how microbial ecosystems respond to a range of distinct perturbations, such as antibiotic exposure, diseases, or other general dynamic properties. This becomes especially challenging due to dozens of complex interactions among microbes and metastable dynamics.

The key to addressing this challenge lies in representing time-evolving graphs constructed from microbiome data as fixed-length, low-dimensional feature vectors that preserve the original dynamics. Therefore, we propose two unsupervised approaches that map the time-evolving graph constructed from microbiome data into a low-dimensional space where the initial dynamic, such as the number of metastable states and their locations, is preserved. The first method relies on the spectral analysis of transfer operators, such as the Perron–Frobenius or Koopman operator, and graph kernels. These components enable us to extract topological information such as complex interactions of species from the time-evolving graph and take into account the dynamic changes in the human microbiome composition. Further, we study how deep learning techniques can contribute to the study of a complex network of microbial species. The method consists of two key components: 1) the Transformer, the state-of-the-art architecture used in the sequential data, that learns both structural patterns of the time-evolving graph and temporal changes of the microbiome system and 2) contrastive learning that allows the model to learn the low-dimensional representation while maintaining metastability in a low-dimensional space.

Finally, this thesis will address an important challenge in microbiome data, specifically identifying which species or interactions of species are responsible for or affected by the changes that the microbiome undergoes from one state (healthy) to another state (diseased or antibiotic exposure). Using interpretability techniques of deep learning models, which, at the outset, have been used as methods to prove the trustworthiness of a deep learning model, we can extract structural information of the time-evolving graph pertaining to particular metastable states.

Zusammenfassungen

Immer mehr Krankheiten stark mit Störungen in der Mikrobiom-Konstitution korrelieren sind, z. B. Fettleibigkeit, Diabetes und einige Krebsarten. Fortschritte in der Hochdurchsatz-omik-Technologie haben es möglich gemacht, das menschliche Mikrobiom und seine Auswirkungen auf die menschliche Gesundheit und Physiologie direkt zu analysieren. Mikrobielle Gemeinschaften werden in der Regel über lange Zeiträume hinweg beobachtet und die Zusammenhänge zwischen ihren Mitgliedern erforscht. In zahlreichen Studien wurden Mikrobiomdaten verwendet, um Krankheitszustände genau zu differenzieren und die Unterschiede in den Mikrobiomprofilen von gesunden und kranken Menschen zu verstehen. Die meisten von Studien konzentrieren sich jedoch hauptsächlich auf verschiedene statistische Ansätze und lassen die Interaktionen zwischen Mikroben und Mikroben einer großen Anzahl von Mikrobiomarten außer Acht, die im Prinzip jedoch die Dynamik des Mikrobioms bestimmen. Um die Reaktionen der Mitglieder einer mikrobiellen Gemeinschaft auf eine Reihe von Störungen wie Antibiotikaexposition, Krankheiten und allgemeine dynamische Eigenschaften zu verstehen, muss der sich im Laufe der Zeit entwickelnde Graph der menschlichen mikrobiellen Gemeinschaften erstellt und analysiert werden. Dies ist aufgrundder zahlreichen komplexen Wechselwirkungen zwischen den Mikroben und der metastabilen Dynamik eine besondere Herausforderung.
Der Schlüssel zur Bewältigung dieser Herausforderung liegt in der Darstellung der sich zeitlich entwickelnden Graphen als niedrigdimensionale Merkmalsvektoren fester Länge, die die ursprüngliche Dynamik erhalten. Daher schlagen wir zwei unüberwachte Ansätze vor, die den aus Mikrobiomdaten konstruierten zeitlichen Graphen in einen niedrigdimensionalen Raum abbilden, in dem die ursprüngliche Dynamik, wie die Anzahl der metastabilen Zustände und deren Orte, erhalten bleibt. Die erste Methode stützt sich auf die Spektralanalyse von Transferoperatoren, wie dem Perron-Frobenius- oder Koopman-Operator, und Graphkerneln. Diese Komponenten ermöglichen es uns, topologische Informationen wie die komplexen Interaktionen von Arten aus dem sich zeitlich entwickelnden Graphen zu extrahieren und die dynamischen Veränderungen des menschlichen Mikrobioms zu berücksichtigen. Außerdem untersuchen wir, wie Deep-Learning-Techniken zur Untersuchung des komplexen Netzwerks mikrobieller Arten beitragen können. Die Methode besteht aus zwei Schlüsselkomponenten: 1) einem Transformer, einer hochmodernen Architektur, die in den sequenziellen Daten verwendet wird und die sowohl strukturelle Muster des sich zeitlich entwickelnden Graphen als auch zeitliche Veränderungen des Mikrobiomsystems erlernt, und 2) kontrastivem Lernen, das es dem Modell ermöglicht, die niedrigdimensionale Repräsentation zu erlernen und gleichzeitig die Metastabilität in einem niedrigdimensionalen Raum zu erhalten.
In dieser Arbeit wird eine wichtige Herausforderung in Bezug auf Mikrobiomdaten angegangen, nämlich die Identifizierung, welche Spezies oder Interaktionen von Spezies für die Veränderungen, die das Mikrobiom von einem Zustand (gesund) zu einem anderen Zustand (krank oder Antibiotikaexposition) erfährt, verantwortlich sind oder davon betroffen sind. Mithilfe von Interpretierbarkeitsverfahren für Deep-Learning-Modelle, die ursprünglich als Methoden zum Nachweis der Vertrauenswürdigkeit eines Deep-Learning-Modells verwendet wurden, können wir strukturelle Informationen eines sich zeitlich entwickelnden Graphen extrahieren, die sich auf bestimmte metastabile Zustände beziehen.

Declaration of authorship

I declare to the Freie Universität Berlin that I have completed the submitted dissertation independently and without the use of sources and aids other than those indicated. The present thesis is free of plagiarism. I have marked as such all statements that are taken literally or in content from other writings. This dissertation has not been submitted in the same or similar form in any previous doctoral procedure. I agree to have my thesis examined by a plagiarism examination software.

Kateryna Melnyk
Berlin, 2024

Acknowledgements

I would like to acknowledge all the people who have supported, encouraged, and motivated me in the completion of this thesis.

First and foremost, I would like to thank my parents, Viktoriia and Oleksandr Melnyk, who have always been my support and encouragement. Thank you for always believing in the importance of a good education and for doing possible and impossible to provide me with it.

I want to thank my supervisor Tim O.F. Conrad who offered me a place in his research group. Thank you for introducing me to the research, always offering your advice and guidance, and your support, motivation, and patience. Further, I would like to thank my colleagues in our group, Sahar Irravani, Kuba Weimann, and Nada Cevetkovic who helped me a lot during my journey and intellectual discussions.

Finally and most importantly, I owe my deepest gratitude to my husband Vlad. Thank you for being proud of me and believing in me when I could not. You have always been by my side during moments of doubt and I am grateful for your emotional support and encouragement throughout this period.

Contents

Table of Contents **vii**

1 Introduction **1**

1.1 Human Microbiome and its Analysis 2

1.1.1 What is the microbiome? 3

1.1.2 Dynamics of microbiome 6

1.2 Related work 10

1.2.1 Statistical analysis of longitudinal data 11

1.2.2 Interactions accountability with mathematical modelling . . . 12

1.2.3 Network perspective on microbiome 14

1.3 Microbiome and complexity reduction 16

1.3.1 Dimensionality reduction 17

1.3.2 Graph representation learning 19

1.3.3 Complexity reduction of microbiome 24

1.4 Thesis Overview 26

2 A graph kernel-based approach for human microbiome analysis **29**

2.1 Transfer operators and microbiome dynamics 33

2.1.1 Reproducing kernel Hilbert space 33

2.1.2 Transfer operators 34

2.1.3 Data-driven methods for approximation of transfer operators . 37

2.2 Graph kernels for microbiome network 41

2.2.1 What is graph kernel? 41

2.2.2 State-of-the-art graph kernels 42

2.3 Novel unsupervised approach for time-evolving graph embedding based on graph kernel and transfer operators 49

2.3.1 Problem Formulation 49
2.3.2 GraphKKE: Graph kernel Koopman Embedding for Human Microbiome Analysis 51
2.3.3 Evaluation 53
2.4 Synthetic data and Results 56
2.4.1 Simulating synthetic data 59
2.4.2 Experiments and Results 60
2.5 Microbiome data and Results 65
2.5.1 Microbiome datasets 66
2.5.2 Experiments and Results 69
2.6 Discussion 73

3 Understanding microbiome dynamics via graph representation learning 75
3.1 Transformer 77
3.1.1 Transformer Architecture 78
3.1.2 Attention mechanism 81
3.1.3 Variations of the Transformer 86
3.2 Contrastive representation learning 89
3.2.1 Problem formulation 89
3.2.2 Contrastive loss 91
3.2.3 Strategies to successful contrastive learning 93
3.3 Novel unsupervised approach for embedding of time-evolving graphs with metastability 94
3.3.1 Problem formulation 94
3.3.2 Input 95
3.3.3 Model architecture 95
3.3.4 The choice of contrastive loss 97
3.3.5 Positional encoding 100
3.3.6 Training and evaluation 101
3.4 Synthetic data and Results 102
3.4.1 Synthetic data 102
3.4.2 Experiments and Results 102
3.5 Microbiome data and Results 108
3.5.1 Microbiome datasets 108

3.5.2 Experiments and Results 109
3.6 Discussion 112

4 Revealing high-impacting modules of microbiome via interpretability 115
4.1 Model interpretation 117
4.1.1 What is an Interpretation? 119
4.2 New Feature Extraction Methods for Microbiome Data 126
4.2.1 Problem formulation 126
4.2.2 LRP-based approach for biomarkers extraction 127
4.2.3 Attention as interpretability 129
4.3 Experiments and Results 132
4.3.1 Experimental setup 132
4.3.2 Results: Synthetic data 134
4.3.3 Results: Microbiome data 139
4.4 Discussion 141

5 Discussion, Conclusion, and Outlook 145
5.1 Discussion and Conclusion 146
5.2 Future Direction 150

Bibliography 153

Chapter 1

Introduction

1.1 Human Microbiome and its Analysis

Approximately every second cell in our body is a microbial cell. We are colonized by a diverse community of bacteria, archaea, and viruses jointly referred to as the microbiome. About 1.5 kg of microbes live almost everywhere on and in the human body as symbionts, e.g., on the skin, in the mouth, or in the gut. They have a strong influence on both their hosts and environments. For example, more and more diseases have been found to strongly correlate with disturbances in the microbiome constitution, e.g., obesity (Hjorth et al., 2018; Menni et al., 2017; Kincaid et al., 2019), diabetes (Qin et al., 2012), and some cancer types (Sánchez-Alcoholado et al., 2020; Gopalakrishnan et al., 2018). Recent studies have also revealed that the gut microbiome also has a huge impact on brain functions and is related to disorders such as Alzheimer's disease (Xu and Wang, 2016). Despite the fact that the microbial communities that live in the human gut have a significant impact on our overall health and well-being, we have a limited understanding of the mechanisms that govern this complex ecosystem.

Most of the studies that develop methods for the analysis of the microbiome or those that make conclusions about processes occurring in the microbiome utilize statistical analyses such as correlation or dissimilarity-based methods. Moreover, inferred conclusions are often made based on methods that can capture only linear correlations. Recently, it has, however, been revealed that although the constitution of the microbiome is constantly changing throughout our lives (in response to environmental factors), a healthy human microbiome can be considered as a metastable state lying in a minimum of some ecological stability landscape (Shaw et al., 2019). Several studies of host-associated and environmental microbiota have revealed cases of complex dynamics, including periodicities, chaos, and alternative stable states (Faust and Raes, 2012). Therefore, the dynamic processes that happen in the microbiota are the essential parts that must be considered when making conclusions about the impact of perturbations on the microbiome composition. Another aspect that is usually ignored during the analysis of the microbiota is the interactions between species presented in the composition of the microbiota. The reason is as follows: 1) Primarily, it is unclear which species interact with each other and how to track their interactions over time, and 2) The absence of methods for analysing such high-dimensional complex data. In the case of the former, more and more stud-

ies have been exploring how to identify interactions between species and how these interactions can be tracked over time. They will be discussed further in the thesis. However, the latter is still under development. Those methods or models that are proposed to capture the interactions between species can only apprehend pairwise interactions ignoring more complex interactions of the microbiome.

We start the chapter with a discussion about what the microbiome is, how microbiome data is collected, and which methods and approaches exist dedicated to the analysis of the microbiome. Considering the challenges of the analysis and the complexity of the microbiome we then explore methods for the reduction of the microbiome complexity. This chapter finishes with the thesis's main objective and research questions.

1.1.1 What is the microbiome?

A microbiome is a composition of bacterial communities present in different environments such as the human body. Common approaches to characterize the human microbiome profile are to use 16S rRNA amplicon sequencing or shotgun metagenomics. Microbiome samples are collected from different body sides such as skin, mouth, or gut. Then, the genome sequence data is obtained through sequencing microbial samples, and the sequence data are processed using programs such as QIIME. The 16S rRNA gene sequences are clustered into groups, the so-called operational taxonomic unit (OTU). Since 90% of microorganisms in the metagenomic data are unknown at present (Song et al., 2018), the OTU grouping can be obtained by using a reference database to align sequences or not having a reference sequence which is a *de novo* approach. Reference-based OTU clustering does not assign sequences that are not aligned to references. Therefore, the two-step OTU clustering approach re-clusters the unassigned sequences in the *de novo* method. When OTU clusters are identified, taxonomic information is assigned to the representative sequences of each OTU to determine the evolutionary relationships or the phylogeny. Despite the introduction of analytical methods that utilize representative sequences of 16S rRNA, such as those presented by Asgari et al., 2018, OTU cluster-based variables remain the most commonly employed input features in the microbiomes and host trait analysis. The overview of the microbiome analysis workflow is shown in Figure 1.1.

These differences in sequence filtering, clustering, and taxonomy assignment chal-

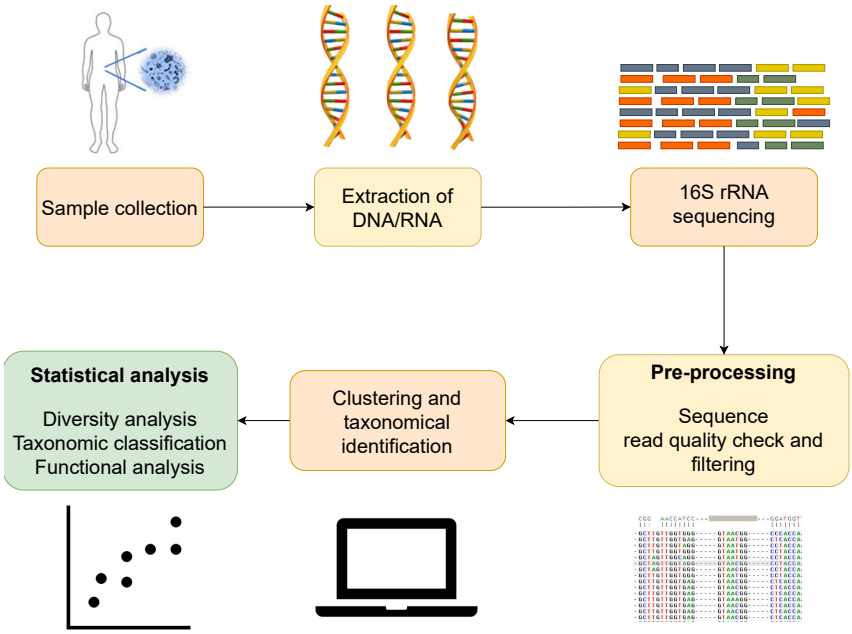


Figure 1.1: Overview of the microbiome analysis workflow. The first step is to gather sample collections from various locations in the body. Once the samples are collected, the laboratory workflow includes DNA extraction, amplification and sequencing. The final sequence data is pre-processed and clustered. Statistical analysis is the last step, which includes a comparison of alpha and beta diversity, correlations between taxa and metadata, and the analysis of microbiome dynamics.

lenge cross-study comparison and reproducibility. Moreover, the human microbiome is incredibly diverse and dynamic, with different populations of microorganisms interacting with each other and with the host in complex ways. Other challenges for current approaches to analysing microbiome data are discussed in (Armstrong et al., 2022):

- *Curse of dimensionality.* Microbiome data is high-dimensional with thousands of species. This leads to a situation where a feature table, commonly referred to as an OTU table, contains a significantly larger number of features compared to the number of samples available. Therefore, this poses a challenge for classical statistical approaches potentially causing overfitting due to the curse of dimensionality.

- *Sparsity.* The microbiome data or a resulting feature table is typically very sparse due to the fact that most microbes are not found in most samples, even of the same biospecimen type. This poses problems for most statistical

methods.

- *Compositionality.* In microbiome research, the compositionality issue arises due to the nature of high-throughput sequencing, where the number of reads obtained from a sample is limited and subject to randomness during sample collection. In traditional ecological data, various species can coexist in a sample, and their counts are often independent. However, in microbiome data, microbial counts are not independent due to the normalization process. The relative abundance of one microbial feature in a sample is inversely related to the abundance of other features in that same sample (Gloor, 2017). Therefore, the classical dimensionality reduction methods, such as principal component analysis(PCA) and other statistical methods such as Pearson correlation usually fail to capture the right correlation in the data. Recently, there have been a lot of efforts made to develop methods robust to this limitation, such as SparCC (Friedman and Alm, 2012) and CCLaso (Fang et al., 2015).

- *Feature importance.* The analysis of microbiome data is largely motivated by finding microbial biomarkers associated with changes in microbial composition. Methods that are not developed particularly for microbiome data but provide a possibility to extract feature importance often fail to account for compositionality.

- *Complex patterns.* Microbiome data is assumed to have clusters and be non-linear. However, many conventional methods for complexity reduction and overall analysis of the microbiome dynamic are designed to capture only linear correlations in the data.

All these challenges and the complexity of microbiome data require the development of approaches that can handle them. The standard procedure is to use different statistical methods for the analysis of microbiome data which is usually based on longitudinal or time-series data. In recent years, mathematical modelling and network analysis have been gaining much attention since they provide a more comprehensive analysis of microbiome data and can tackle some of the above-mentioned challenges. Later, we will dive deeper into some existing methods for the analysis of microbiome data.

1.1.2 Dynamics of microbiome

The temporal changes in host-associated microbial communities are of growing interest due to their relevance to human health. Normally, human microbiota remains stable for months, and possibly even years (David et al., 2014; Faith et al., 2013). However, some research suggests that microbial communities are in fact not static over time. A change in the community state can be triggered by changes in external conditions (e.g. diet), by a direct modification of the microbial community (antibiotic exposure), or by a transient perturbation that pushes the system into an alternative state. Healthy microbiome communities however tend to evolve towards a stable composition. Therefore, several studies apply the idea of a potential or energy landscape to microbiome research. Potential landscape formalism considers a high-dimensional phase space, in which coordinates represent system states, and system dynamics correspond to trajectories through phase space. For instance, the approach proposed in (Chang et al., 2020) is based on this idea of potential landscapes. Although one can argue that the idea based on the potential landscape may not be directly applicable to microbiomes due to the open nature of the system, the authors claim that the local maxima of the data density could form a basis for inferring the feasibility of metastable states of microbiome composition. To implement it, the authors used the Mapper algorithm which represents the underlying data distribution in a metric space as an undirected graph. In this graph, each vertex comprises a non-exclusive subset of data points spanning a phase space. An edge is drawn between a pair of vertices that share at least one data point. They experimented with three microbial time series datasets of human gut microbiomes.

The metastability phenomena is a big part of the dynamical systems theory and it goes back to the work of Freidlin and Wentzell in the early 1970s. They mainly considered the setting of finite dimensional dynamical systems perturbed by weak additive noise (Bovier, 2006). In terms of microbiome research, the idea of applying a stability landscape paradigm to the microbiome dynamics was taken from classical ecological theory. The definition of stability or metastability from dynamical system theory may be different from how it is understood in the microbiome field. In the dynamical system theory, metastability can be observed when for short timescales, the system appears to equilibrate, but at larger time scales, undergoes some transitions from one metastable state to other metastable states (Bovier, 2006). This phenomenon occurs in dynamical systems of various structures, including systems

with vector-valued states and systems represented as time-evolving graphs. As we propose to construct a time-evolving graph from the microbiome data in this thesis, the metastability in time-evolving graphs means that the graph structure is stable for a relatively long time (up to small perturbations) before the system undergoes a critical transition — e.g. when it reaches a tipping point — and shifts to a different metastable state (the structural patterns of the time-evolving graph dramatically change).

Since stability is a focus area of many works in ecology and other relative domains, there have been many definitions of stability proposed. (Gonze et al., 2018) gives a good overview of four different types of "stability", which are also commonly used in research on the microbiome data:

- *Linear asymptotic stability.* If the composition of a microbial community returns to its initial (steady) state following a perturbation, this state will be stable. On the contrary, if the microbial composition diverges from its initial state, this state is unstable.

- *Persistence and permanence.* Both terms imply that irrespective of the size of the perturbation, the system always maintains its initial composition of species. However, permanence means that if any species' density is approaching zero, it will begin to grow. So, theoretically, no species can become extinct.

- *Temporal stability and robustness.* If the system of species tends to remain constant over time or across parameter changes, which usually describe environmental changes, it will be considered more stable. On the other hand, if the abundance of some species is sensitive to parameter changes, the system will be considered less stable. This concept of stability is a measure of robustness to noise and to parameter values.

- *Structural stability.* This type of stability implies that the coexistence between several species is maintained over a larger range of parameter values.

In this sense, linear asymptotic stability is the closest definition to the metastability defined in the dynamical system theory. Temporal stability and robustness are defined with regard to different parameters in the equations used to describe the dynamic of the microbiome. Persistence, permanence, and structural stability imply the initial conditions of a dynamical system supported by biological justification presented in different studies.

Other works that use the stability landscape to study the microbiome dynamics include (Lahti et al., 2014; David et al., 2014; Hsiao et al., 2014; Shaw et al., 2019; Zaura et al., 2015). The study of (Lahti et al., 2014) utilized potential analysis to detect alternative states in intestinal bacteria. This approach is based on the theory of stochastic dynamical systems. They assume that an underlying stochastic dynamical system of the microbiome can be described with the following equation

$$dz = -U(z)dt + \sigma dW_t, \tag{1.1}$$

where $U(z)$ is the potential function, z is the state variable (the microbe $\log_{10}(abundance)$), σ is the noise level and dW is a Wiener process. In particular, the authors proposed to employ a potential function:

$$U = -\frac{\sigma^2 log(p_d)}{2} \tag{1.2}$$

where p_d is the empirically estimated probability density function of the state variable z. The minima of the potential function corresponds to a stable state of the system. They further estimate the probability density of bacteria abundance with a Gaussian kernel. They also used the number of local minima of the potential function to quantify the number of distinct states.

The work from (Shaw et al., 2019) follows the same approach as the previously mentioned study by considering the gut microbiome as lying in a stability landscape. They derive a mathematical model to investigate whether it could provide insights into the effect of antibiotics on the gut microbiome. The visualization of their approach is shown in Figure 1.2. They treat the local stability landscape as a harmonic potential, with a 'restoring' force proportional to the displacement x from the equilibrium position $(-kx)$. They also consider the presence of a "frictional" force acting against the direction of motion $-bx$. Finally, the system is equivalent to a damped harmonic oscillator with the following equation of motion:

$$\frac{d^2x}{dt^2} + b\frac{dx}{dt} + kx = 0. \tag{1.3}$$

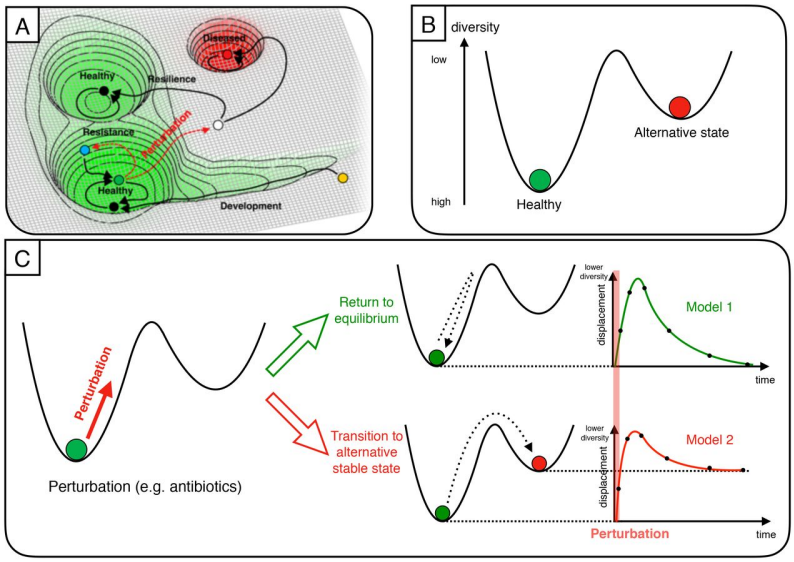


Figure 1.2: The figure is taken from (Shaw et al., 2019) and illustrates the idea of a stability landscape of the gut microbiome. **(A)** The healthy human microbiome is thought to rest in a state of equilibrium within the stability landscape of all possible states of the microbiome. Different perturbations can displace it from this equilibrium value into alternative states. **(B)** It is assumed that there are only two states: the healthy baseline and an alternative stable state. **(C)** Perturbation to the microbiome (in this case by antibiotics) is modelled as an impulse.

Since antibiotic treatment may lead not just to displacement, but also to new equilibria, the authors propose a solution to Equation 1.3:

$$x_2(t) = \frac{De^{\phi_1}e^{\phi_2}}{e^{\phi_2} - e^{\phi_1}} \cdot \left(e^{-e^{\phi_1}t} - e^{-e^{\phi_2}t}\right) + A \cdot \left(1 - e^{-e^{\phi_1}t}\right), \tag{1.4}$$

with the initial conditions $x(0^+) = 0$ and $\dot{x}(0^+) = D$, $b = e^{\phi_1} + e^{\phi_2}$, $k = e^{\phi_1+\phi_2}$. D is an impulse of magnitude or how strong the perturbation and A is a new equilibrium state. Afterwards, the proposed model is validated with an empirical dataset. They base their experiment on data from (Zaura et al., 2015), where 30 individuals received a ten-day course of either a placebo, ciprofloxacin or clindamycin. They found support for a long-term transition to an alternative community state one year after the antibiotic exposure. Their results also imply that a single treatment of antibiotics not only reduces the diversity of the gut flora for up to a year but also potentially alters its composition. (David et al., 2014) studied gut and salivary microbiota dynamics of two individuals over the course of one year. One result of the

study suggests that overall microbial communities are stable for months. However, rare events such as travelling from the developed to the developing world lead to a nearly two-fold increase in some bacteria taxa. Another event, namely enteric infection, resulted in the permanent decline of most gut bacteria taxa. (Hsiao et al., 2014) conducted a time-series metagenomic study of the faecal microbiome collected during the acute diarrhoeal and recovery phases of cholera in a cohort of Bangladeshi adults living in an area with a high rate of cholera. The study noted taxa that consistently changed in abundance between the start and end of the diarrhoea phase. Moreover, they identified a set of bacterial species that strongly correlate with a process in which the perturbed gut bacterial community in adult patients with cholera is restored to a configuration found in healthy Bangladeshi adults.

Modelling the microbiome as a stability landscape provides new insights into the dynamical properties of the microbiome and associations between the microbiome constitution and different external or internal perturbations such as diseases or antibiotic exposure. Yet, this also presents further challenges with the analysis of the microbiome dynamics, especially if we also aim at accounting for the multiple interactions between species. Given the challenges with microbiome data, thousands of complex interactions and its metastable behaviour we need to develop approaches that can deal with all these hardships to extract valuable insights from the microbiome data.

1.2 Related work

Over the years, approaches for analysis of the microbiome data have been extensively developing. These approaches mainly originate across different domains with different goals and the types of data used in the analysis of microbiome dynamics. In this section, we will give an overview of different methods used to analyze the microbiome dynamics. We can divide them into several groups based on the data types and properties of the microbiome dynamics which these methods are able to capture. The first category is a statistical analysis of longitudinal or time-series data that still aims at extracting insights from the microbiome data but ignores multiple interactions and temporal changes. Later, it was realized that taking into account the interactions between species is crucial for understanding the microbiome dynamics. Therefore, there have been approaches proposed based on mathematical models such

as the Lorka-Volterra model, that consider pairwise interactions, and other external associations with the microbiome. However, afterwards, it was discovered that more complex interactions than just pairwise interactions are inherent to the microbiome which resulted in the development of approaches that utilize network analysis such as probabilistic graphical models or dynamical graphs.

1.2.1 Statistical analysis of longitudinal data

High-throughput experimental methods enable comprehensive analysis of the microbiome over time which opens up opportunities to provide insights into fundamental questions about microbiome dynamics. These methods facilitate the creation of a lot of longitudinal and time-series data which have the property of order and time. A broad overview of studies and open questions of the analysis of the microbiome dynamics from longitudinal data is presented in (Gerber, 2014). It is worth mentioning that in the microbiome community, the terms longitudinal data and time-series data are often used interchangeably. However, the big difference is that in a time series we measure the overall change in the measurement over time while in a longitudinal analysis, the researcher conducts several observations of many subjects at different time points.

The main concern with the analysis of longitudinal data is that a lot of studies attempt to answer all questions related to microbiome dynamics with simple analytical tools. However, if these tools ignore temporal aspects of data, the results can be extremely misleading (Gerber, 2014). (Caporaso et al., 2011) assess the degree of variation over time in the microbiota of different body sides of healthy adults based on principal coordinates analysis of unweighted UniFac distances between communities, a measure of community dissimilarity based on OTU presence or absence. (Gajer et al., 2012) investigate the changes in the vaginal microbiota over time based on a longitudinal study. They used the Jenson-Shannon metric to measure dissimilarity between community states. After that, to analyze the temporal dynamics of the vaginal microbiota they used a linear mixed-effects model.

Some studies explore the dynamic response of the microbiome to internal perturbations using statistical analysis. (Dethlefsen and Relman, 2011) recruited three healthy subjects who received two 5-day courses of oral ciprofloxacin during a 10-month study period. To measure similarities and dissimilarities between the membership of the distal gut microbial communities between the first and second courses

of antibiotics, researchers used Principal Coordinate Analysis (PCoA) of unweighted UniFrac distances. (Pérez-Cobas et al., 2013) studied the effects of antibiotics on the microbiome of a single human subject using a multi-omics approach. After generating time series from 16S rRNA gene sequences, they conducted a statistical analysis using Principal Component Analysis, and Spearman's correlation analysis and identified the changes in the microbiome constitution after the antibiotic treatments.

These studies have a common pattern in the analysis: they start with sample collection, then DNA extraction, sequencing, and taxonomic classification and they finish with statistical analysis to make conclusions about the microbiome dynamics and differences in samples. Some of the above-mentioned studies include metagenomic sequences and proteomic analysis but still, the last step is statistical analysis. In general, the analysis of longitudinal microbiome data presents certain challenges. One challenge lies in a poorly designed longitudinal study which in an ideal case requires close collaboration between biologists and computational experts. Another challenge is directly connected to the desire to obtain "quick answers" with simple analytical tools or statistical methods. If these methods or tools ignore the temporal aspect of longitudinal or time-series data, it can result in incorrect conclusions (Gerber, 2014).

1.2.2 Interactions accountability with mathematical modelling

Studies that use statistical analysis for inferring microbiome dynamics omit the large variety of complex microbe-microbe and host-microbe interactions. In order to tackle this problem, many studies in the microbiome domain have been making attempts to describe the microbiome dynamic with various mathematical models. The most popular model is the Lotka-Volterra model and its variations (Momeni et al., 2017; Stein et al., 2013; Fisher and Mehta, 2014). The model is based on inferring the network of underlying pairwise interactions between taxa by calculating the inverse covariance matrix from time series data. The model can be used to predict 1) how changes in the abundance of one microbial taxon affect the abundance of other taxa and 2) how the overall stability of the microbiome is impacted by different perturbations.

(Stein et al., 2013) employed a dynamical system model for analysing microbiome time-series data. They used a model based on generalized Lotka-Volterra (gLV) non-

linear differential equations. Their model uses the following equation, which extends the gLV equations to include terms for externally applied perturbations such as antibiotic treatment:

$$\frac{dx_i(t)}{dt} = \mu_i x_i(t) + x_i(t) \sum_{j=1}^{D} \alpha_{ij} x_j(t) + x_i(t) \sum_{k=1}^{P} \beta_{ik} c_k(t), \tag{1.5}$$

where $x_i(t)$ is a concentration of species i at time t, μ_i represents its intrinsic growth-rate, α_{ij} is the effect if species j on species i, and β_{ik} represent the effect of time-dependent perturbation $c_k(t)$ on species i.

While the previous study does not account for the compositionality of the microbiome and describes dynamics in terms of absolute densities of taxa, (Joseph et al., 2020) proposes a new approach to modelling microbial dynamics using the Compositional Lotka-Volterra framework. This approach takes into account the compositionality of the microbiome by using the additive log-ratio transformation in the gLV equation 1.5

$$\frac{d}{dt} \log \frac{\pi_i(t)}{\pi_D(t)} \approx \bar{\mu}_i + \sum_{j=1}^{D} \bar{\alpha}_{ij} \pi_j(t) + x_i(t) \sum_{k=1}^{P} \bar{\beta}_{ik} c_k(t),$$

where $\pi_i(t) = \frac{x_i(t)}{N(t)}$, $N(t) = \sum_{j=1}^{D} x_j(t)$, $\mathcal{E}[N(t)] = 1$, P is a number of external perturbations, D is the chosen scale of the concentration of taxon i, and $\bar{\mu}_i = \mu_i - \mu_D$, $\bar{\alpha}_{ij} = \alpha_{ij} - \alpha_{Dj}$, $\bar{\beta}_{ik} = \beta_{ik} - \beta_{iD}$ are relative growth rates, relative interactions, and relative external effects respectively.

Although mathematical modelling methods, such as the Lotka-Volterra model, can be valuable tools for assessing the stability of bacterial communities over time, they may not be suitable for handling microbiome time series data that are both temporally sparse and high-dimensional (Lugo-Martinez et al., 2019). The majority of these methods also remove any taxa that have a relative abundance profile containing a zero entry (Lugo-Martinez et al., 2019). Moreover, since this approach is based on pairwise interactions, it ignores more complex interactions. We can summarize the limitation of this model in the following list provided by (Gonze et al., 2018):

- It only describes pairwise interactions and thus fails to capture more complex interactions between microbes.
- Interaction strengths are not supposed to change over time.

- Interactions are assumed to be bilinear meaning that the growth rate of a species will change proportionally to the abundance of its connected neighbour.
- It does not take into account environmental factors such as variable temperature or antibiotic exposure.

All these limitations might be acknowledged by considering more complex interactions in the microbiome constitution that are changing over time. For this reason, the next section will give insights into how network analysis has been used to study microbiome dynamics.

1.2.3 Network perspective on microbiome

Methods discussed above mainly focus on statistical constitution analysis (longitudinal analysis or time-series analysis), omitting the large variety of complex microbe-microbe and host-microbe interactions, or on mathematical modelling that only accounts for pairwise interactions. However, it is known that the microbiome contains a lot of complex interactions that have to be considered. That is why, the network perspective gives a more comprehensive approach to the analysis of microbiome dynamics than mathematical modelling with Lotka-Volttera, especially the analysis based on statistical methods. The main focus from the network perspective of the microbiome is on how to construct a network from the microbiome data, especially a network for tracking changes over time, which is crucial in the analysis of microbiome dynamics.

Several methods are available for constructing microbial networks, ranging from permutation tests and correlation to approaches dealing with compositionality or multiple factors influencing taxon abundance (regression). The simplest approach for building microbiome networks is based on similarity and dissimilarity measures. These methods use a pairwise dissimilarity index such as the Bray–Curtis or Kullback–Leibler indices. Other methods are more complex and include more substantial approaches to constructing networks from microbiome data.

Correlation networks

Correlation-based techniques for constructing microbiome networks detect significant pairwise interactions between OTUs using a correlation coefficient such as Pearson correlation coefficient or Spearman's nonparametric rank correlation coefficient. The

basic assumption of this approach is that a non-random pattern is shaped by ecological processes driving coexistence or exclusion. However, the nature of this pattern is usually not defined (Riera and Baldo, 2020). (Faust and Raes, 2012) analyzed ecological interactions among bacteria in the human microbiome using a co-occurrence network. The network was constructed using similarity and dissimilarity measures and generalized boosted linear models which describe predictive relationships.

Pearson and Spearman correlation coefficients are not however robust to compositionality. Hence, several construction methods have been proposed to deal with the compositionality of microbiome data. Sparse Correlations for Compositional Data (SparCC) is one of these methods. SparCC was developed by (Friedman and Alm, 2012) and it was proven to work accurately on simulated and real-world datasets. This method employs log transformations which are required to eliminate zero values from datasets or they can be replaced with small values known as pseudocounts. Another method for inferring correlations from compositional data is Correlation inference for Compositional data through Lasso (CCLasso) (Fang et al., 2015). It uses the least squares with an L1 penalty after log ration transformation for raw compositional data to identify the correlations among microbes. Molecular Ecological Network Analysis Pipeline (MENAP) (Deng et al., 2012) is a random matrix theory-based method that is developed for tackling the issue of arbitrary choice of thresholds used to include or exclude interaction from the networks.

Probabilistic graphical models (PGMs)

PGMs, with Bayesian and Markov networks being the most popular graphical models, bring together probability theory and graph theory to deal with uncertainty and complexity. Bayesian networks are considered directed graphical models, while Markov networks are considered undirected graphical models. Directed graph models can be represented by a graph with its vertices serving as random variables and directed edges serving as dependency relationships between them. In the case of Bayesian networks, the directed edge represents conditional distributions. For example, if the values of the vertices are binary, the conditional distributions may be Bernoulli distributions. In the case of continuous values, the conditional distributions may be Gaussian. Markov random fields (MRFs) or Markov networks also use a graph to describe dependencies between random variables, but in this case, MRFs use undirected edges instead of directed edges. This difference implies what type of

relationships each of these methods can capture. Bayesian networks are better suited for modelling causal relationships, while MRFs can be used for modelling spatial and temporal dependencies. Depending on whether there is temporal information or not, one can use static or dynamic PGMs.

In terms of microbiome dynamics, PGMs can be used to model relationships between different microbial taxa. Edges between OTUs represent symmetric undirected associations. These models can capture the underlying stochastic processes that govern the dynamics of the microbiome. PGMs also can be used to model the interactions between the microbiome and host factors which allows identifying key host-microbe interactions that influence the composition and function of the microbiome (Hernández-Rocha et al., 2021; Lugo-Martinez et al., 2019).

Other methods for constructing networks include *regression-based methods*. The regression-based methods are based on multiple regression analysis to infer the abundance of one species from the abundance of another species. Unlike correlation networks or co-occurrence-based methods, regression analysis is able to capture more complex forms of interactions. As with all machine learning methods, regression-based methods suffer from overfitting, the curse of dimensionality, and the interpretation of results may become more difficult. For example, the predicted links might not always mean that there is a biological association between species (Layeghifard et al., 2017).

Network analysis and the Lotka-Volterra model are two commonly used approaches for studying the dynamics of microbial communities in the human microbiome. Both approaches can give insights into the interactions between different microbial taxa and how these interactions contribute to the overall stability of the microbiome. Although these approaches are two different complementary approaches, they can be combined to provide a more comprehensive understanding of the dynamic of microbial communities in the human microbiome.

1.3 Microbiome and complexity reduction

To address the challenge posed by the high dimensionality of microbiome data, numerous studies employ various dimensionality reduction techniques to reduce the complexity of data. Therefore, dimensionality reduction is one of the main steps used in the analysis of the microbiome data. In this Section, we will explore different

dimensionality reduction methods that are used both in the microbiome community and in other domains.

1.3.1 Dimensionality reduction

Dimensionality reduction is a technique used in machine learning to reduce the number of features or variables in a dataset, while still retaining as much information as possible. This can be useful for a variety of reasons, such as reducing the computational complexity of a model, improving the interpretability of results, and mitigating the curse of dimensionality. Moreover, the dimensional reduction methods can also be useful for visualizing high-dimensional data and identifying patterns or clusters. One of the most common dimensionality reduction techniques is principal component analysis (PCA).

PCA is a classical dimensional reduction method used in data analysis and machine learning. The idea of PCA is to find a new set of variables, the so-called principal components, that are linear combinations of the original variables, and that capture the most important patterns and variations in the data. PCA is limited by certain assumptions, namely, it can capture linear correlations between the features but fails when this assumption is violated. Moreover, traditional PCA is not suitable for large datasets or online computing. Several variations of PCA have been developed to address these challenges.

- *Kernel PCA* (Schölkopf et al., 2005) is a nonlinear dimensionality reduction technique. The basic idea behind it is to first map data to a higher-dimensional space with a kernel function with more separable data. Then, the principal components are calculated in this new high-dimensional space. However, kernel PCA can be computationally expensive and requires tuning of the kernel parameters, which is time-consuming and prone to errors.

- *Sparse PCA* (Zou et al., 2006; Shen and Huang, 2008) produces sparse principal components by introducing a sparsity constraint on the coefficients of the eigenvectors. This variation is useful in cases where only a small subset of the original features are relevant for explaining the variance in the data.

- *Multi-Block PCA* (Bair et al., 2006) can handle data that is inherently divided into multiple blocks or groups by decomposing each block separately and then combining the results.

- *Incremental PCA* (RossD, LimJ et al., 2008) which allows for the computation of principal components in a streaming or online fashion. Instead of computing all principal components all at once on static data, incremental PCA updates the principal components incrementally as new data is observed. The main advantage of this PCA variation is that it can handle very large datasets that cannot fit into memory. One limitation of Incremental PCA is that it may not produce the same results as traditional PCA if the data distribution changes over time.

- *Robust PCA* (Wright et al., 2009; Candès et al., 2011) was developed to tackle challenges such as the presence of outliers and noise in the data. This is done by decomposing the data into a low-rank component and a sparse component that captures the outliers.

- *Non-negative PCA* (Lee and Seung, 1999; Hoyer, 2004) produces non-negative principal components which can be useful in cases where the data is naturally non-negative, such as image data or spectral data. The non-negative principal components are achieved by imposing non-negativity on the loading matrix, which represents the contribution of each variable to each non-negative principal component.

- *Probabilistic PCA* (Tipping and Bishop, 1999; Lawrence and Hyvärinen, 2005) outputs the principal components as a probabilistic distribution which allows for the incorporation of prior knowledge and uncertainty about the data. The probabilistic formulation of PCA is obtained from a Gaussian latent variable model.

Each of these variations has its own advantages and disadvantages. Choosing which algorithm to use depends mainly on the specific requirements of the problem at hand, such as data type or computational feasibility.

Another dimensionality reduction approach is t-distributed stochastic neighbour embedding (t-SNE) (Maaten and Hinton, 2008; Van Der Maaten, 2014). Even though t-SNE was initially proposed for the visualization of high-dimensional data in a lower-dimensional space, typically 2D or 3D, (Van Der Maaten, 2009) later introduced a parametric t-SNE as an unsupervised dimensionality reduction technique. The idea is to represent each data point as a high-dimensional probability distribution and then find a low-dimensional embedding that preserves the pairwise simil-

arities between the data points as much as possible. This is achieved by minimizing the Kullback–Leibler divergence between the high-dimensional and low-dimensional distributions.

Other dimensionality reduction techniques are non-negative matrix factorization, autoencoders, linear discriminative analysis etc. Non-negative matrix factorization is a technique that factorizes a non-negative matrix into two lower-dimensional non-negative matrices. Autoencoders are neural networks that have a bottleneck which is a compressed knowledge representation of the original input. These two methods are often used in image and text processing.

1.3.2 Graph representation learning

If we consider the network perspective on the microbiome, then graph representation learning may be a better choice for the reduction of microbiome complexity. Therefore, since we will use not only the "dimensional reduction" term, which has been extensively used in microbiome research but also "graph representation", we believe it is worth explaining what graph representation is and shedding light on the difference between these two terms in the context of graph-structure data.

Graph representation learning is a process of learning a low-dimensional vector representation, the so-called *embedding*, of nodes in a graph while preserving their structural information. Note that graph representation learning can be done not only for nodes but also for the entire graph. In this case, the task is more challenging since it requires capturing the global properties of the graph such as its overall topology, and community structure, while still maintaining a low-dimensional representation. Graph representation learning has gained a lot of attention in recent years due to the wide range of applications such as social network analysis, and recommendation systems (Ying et al., 2018a; Wang et al., 2019; Wang et al., 2020a)

We can broadly categorize methods for graph representation learning into semi-supervised or unsupervised methods and methods for static or time-evolving (dynamic) graphs. A good overview of the current state of methods for time-evolving and static graph representation techniques can be found in (Kazemi et al., 2020; Barros et al., 2021; Cui et al., 2019; Zhang et al., 2020a). The most recent survey on both time-evolving graphs and static graphs is presented in (Khoshraftar and An, 2022).

Static graph representation

Approaches for static graph representations can be classified into two categories – those that learn the representation of nodes and sub-structures of the graphs. The first category tends to encode nodes of the graph in a low-dimensional space such that their topological properties are reflected in the new space (node2vec (Grover and Leskovec, 2016), DeepWalk (Perozzi et al., 2014)). Most studies are focused on node representation learning, and only a few learn the representation of the whole graph (graph2vec (Narayanan et al., 2017)).

Representing an entire graph using node embeddings is challenging because pooling a graph into a vector representation usually introduces an extreme information bottleneck. Simple approaches to this problem aggregate all node embeddings (e.g., sum or average) or create a "master node" that is connected to all the other nodes in the graph. Recent graph pooling operations can learn hierarchical representations that significantly reduce the size of a graph. For instance, the authors in (Ying et al., 2018b) propose a differentiable graph pooling module called DiffPool that learns to assign nodes to clusters, resulting in a gradual pooling of the graph. Further improvements to hierarchical pooling can potentially reduce the number of parameters and the complexity of the operator, and they might increase the overall performance. For instance, SAGPool (Lee et al., 2019) proposes the self-attention mechanism using graph convolution in the graph pooling, and HGP-SL (Zhang et al., 2019) introduces a structure learning mechanism. In contrast to these approaches, we leverage the self-attention mechanism of a Transformer (Vaswani et al., 2017) in our model and we add a master node that attends to every node in the graph. This simple graph pooling only marginally increases the number of parameters and the computational complexity of the model. Furthermore, we use the initial representation of the master node to inject the topological information of the time-snapshot graph at a previous time step.

Dynamic graph representation

Representing a time-evolving or dynamic graph in a low-dimensional space is an emerging topic that is still being investigated. Some methods are the extension of the methods for static graphs and are based on matrix factorization (Yu et al., 2017; Zhang et al., 2018b; Zhu et al., 2016). These methods factorize the similarity matrix to generate node embeddings over time. Among recent approaches, DynGEM (Goyal

et al., 2018) uses the learned representation from the previous time step to initialize the current time step representation. Such initialization keeps the representation at the current time step close to the learned representation at the previous time step. The extension of the previous method is dyngraph2vec (Goyal et al., 2020), where authors have made it possible to choose the number of previous time steps that are used to learn the representation at the next time step. Moreover, dyngraph2vec uses recurrent layers to learn the temporal transitions in the graph. Several approaches perform random walks on each snapshot of a time-evolving graph (Mitrovic and De Weerdt, 2018; De Winter et al., 2018; Zhou et al., 2019), however generating random walks for every timestamp is too computationally expensive. Therefore, other approaches have been proposed that first generate embeddings for the initial time step using static random walk, and then they gradually update node representation (Mahdavi et al., 2018; Heidari and Papagelis, 2020; Sajjad et al., 2019; Khoshraftar et al., 2019).

Graph Neural Networks

Graph Neural Networks are a class of machine learning methods for processing data that can be represented as graphs. GNNs aim to generalize the concept of convolutional neural networks to graph data by introducing an aggregation function that aggregates information from neighbouring nodes. All GNNs usually utilize the following fundamental layers: permutation equivariant, local pooling and global pooling. The permutation equivariant layer maps a representation of a graph into an updated representation of the same graph. Usually, it is implemented by pairwise message passing between nodes of the graph, which implies updating node representation through aggregating messages received from its immediate neighbours. Local pooling is done similarly to convolutional neural networks and is used to increase the receptive field of a GNN. Global pooling provides a fixed-size representation of the whole graph. We further discuss some most commonly used types of GNNs. According to (Wu et al., 2020), they can be divided into the following main groups:

- *Recurrent graph neural networks:* aims to learn node representation with recurrent neural network architectures. The main assumption in Recurrent GNNs is that nodes in the graph constantly exchange information with their neighbours. They apply the same set of parameters recurrently over nodes in a graph to extract high-level node representation. The earlier research on this

type of GNNs includes (Sperduti and Starita, 1997; Micheli et al., 2004; Li et al., 2015).

- *Convolutional graph neural networks:* use a specific type of aggregation function (i.e. graph convolutional) to update node representation. The main difference with recurrent graph neural networks is that they use multiple graph convolutional layers to extract high-level node representations. Two major types of GCNs are *Spatial Graph Convolutional Networks* (Danel et al., 2020) and *Spectral Graph Convolutional Networks*. Usually, models based on spatial convolution are chosen over models based on spectral convolution due to efficiency, generality and flexibility (Wu et al., 2020). Spatial convolution improves on the classical GCN (Kipf and Welling, 2016a) by taking into account the ordering of node neighbours and it extracts properties of a target node based on its k local neighbours. *GraphSAGE* (Hamilton et al., 2017) is an example of spatial GCN. It leverages node feature information to generate inductive node embeddings for previously unseen data. They train a set of functions that learn to aggregate feature information from a node's local neighbourhood by incorporating a different number of hops or search depths. Another example of spatial convolution is Graph Attention Network (Velickovic et al., 2017). It uses an attention mechanism to selectively attend to different nodes in the graph using masked self-attentional layers. Attention-based node embedding aims to assign an attention weight to neighbourhood nodes of a target node. The attention mechanism allows efficient and scalable computation of node representation from neighbours most relevant for a particular node. An improvement to Graph Attention Network is done in Gated Attention Networks (Zhang et al., 2018a), where a self-attention mechanism computes an additional attention score for each attention head. Message Passing Neural Networks (Gilmer et al., 2017) outlines a general framework of spatial-based GNNs. The basic idea is to consider graph convolutions as a message-passing process in which the latent representation of a target node is done by passing messages between nodes along edges. The architecture consists of two key components: a message function and a node update function. The former takes the latent representation of two nodes and computes a message to be sent from one node to the other. The latter takes the current latent representation of a node and messages it has received from its neighbours and computes a new latent

representation of that node. *Spectral Convoluiton* performs an eigendecomposition of the Laplacian matrix of the graph. An analogy of this approach is PCA where the eigendecomposition of the feature matrix is performed to understand the variance of data. However, in the case of spectral convolution smaller eigenvalues explain the structure of the graph better. Examples of methods based on spectral convolution include (Bruna et al., 2013; He et al., 2022; Kipf and Welling, 2016a)

- *Graph autoencoder:* is a set of unsupervised learning methods which map nodes or graphs into a latent feature space and then decode graph information from latent representation. These methods are used for learning graph embeddings and graph generative distributions. This category includes models such as Structural Deep Network Embedding (SDNE) (Wang et al., 2016), which uses a stacked autoencoder to learn the node's first-order proximity and second-order proximity. Variational Graph Autoencoder (VGAE) (Kipf and Welling, 2016b) is a framework based on a variation autoencoder which is able to learn interpretable latent representation for undirected graphs.

- *Spatial-temporal graph neural networks:* All methods discussed above are mainly meant to work with static graphs. However, many real-world applications utilize the concept of dynamic or time-evolving graphs. Spatial-temporal GNNs are built in a such way that they are capable of capturing temporal patterns alongside spatial patterns. Most RNN-based approaches are able to capture spatial-temporal dependencies (Zhang et al., 2018a; Seo et al., 2018; Li et al., 2017; Wu et al., 2019). EvolveGCN (Pareja et al., 2019) applies the graph neural network for static graphs to dynamic graphs by introducing a recurrent mechanism to update the network parameters. The authors focus on the graph convolutional network and incorporate a recurrent neural network to capture the dynamics of the graph. DynSAT (Sankar et al., 2020) which learns a dynamic node representation by considering topological structure (neighbourhood) and historical representations following the self-attention mechanism.

Overall, dimensionality reduction and graph representation learning are both techniques used in machine learning to reduce the complexity of data, but they operate in different domains and have different goals. Dimensionality reduction is

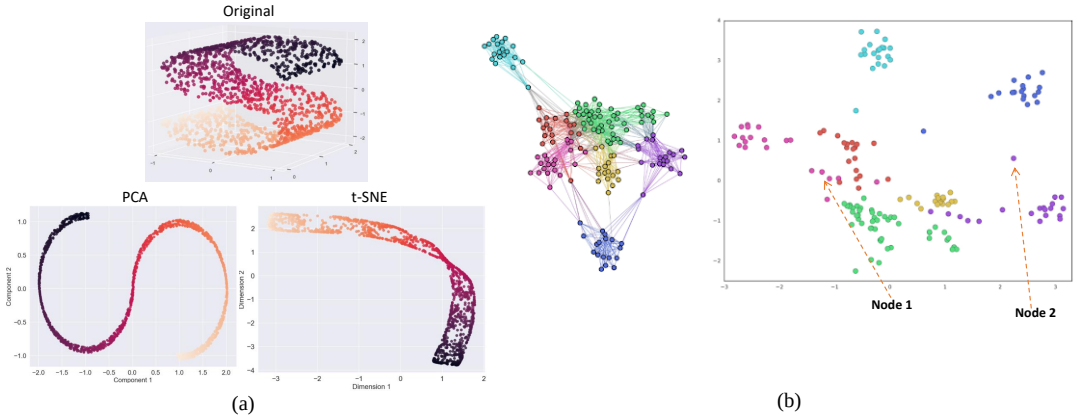


Figure 1.3: The visual comparison of dimensionality reduction **(a)** and graph representation **(b)**. The former aims at reducing the number of dimensions while preserving the most important information. The latter, in turn, is objected to learning a mapping that embeds nodes, or entire graphs as points in a low-dimensional vector space while considering relationships between nodes.

typically applied to datasets that are represented as vectors in a high-dimensional space, where each dimension represents a feature or variable. Graph representation learning, on the other hand, is concerned with learning a graph embedding that preserves some notion of structure or connectivity between the nodes. However, graph representation learning can be also applied to learning an embedding of the whole graph, instead of a single node. In this case, dimensionality reduction and graph representation learning have the same objective. In Figure 1.3 we visually illustrate the difference between dimensionality reduction (PCA and t-SNE) and the classical understanding of graph representation learning (the method is from (Epasto and Perozzi, 2019).

1.3.3 Complexity reduction of microbiome

Microbiome data is high-dimensional data with numerous OTU cluster variables. Therefore, dimensionality reduction has become a common choice to facilitate the exploration and visualization of community similarity and distribution across sample populations.

In the microbiome community, PCA and principal coordinate analysis (PCoA) are two commonly used dimensionality reduction methods. However, PCA is generally better suited for identifying the features that explain the most variance in the data

and despite its limitations and challenges associated with microbiome data discussed in Section 1.1, it has widely been used in microbiome analysis. For example, it was used to derive complex gut microbiota patterns associated with diet and body composition (Leong et al., 2020). (Stanaway et al., 2023) applied PCA to identify microbiome clusters and test their association with pesticides. As it was mentioned above, compositionality and sparsity induce PCA to produce misleading results on microbiome data (Morton et al., 2017). (Morton et al., 2019) proposed to adopt *Robust PCA* to microbiome data which has shown a lot of successful applications in various biological contexts (Beauchamp-Walters et al., 2023; Bali et al., 2021; Parbie et al., 2021). The generalized version of this method proposed in (Martino et al., 2021), called compositional tensor factorization, has shown that it is possible to detect microbial changes associated with specific phenotypes across both time and space.

PCoA is mainly applied to microbiome data and used to analyze and display distance matrices, which characterize how different or similar microbial communities are based on their taxonomic or functional profiles. In order to preserve the original distances as much as possible, PCoA converts the high-dimensional distance matrix into a lower-dimensional space, usually a two- or three-dimensional space. Some studies have revealed microbiome-disease associations using PCoA (Campbell et al., 2020). However, both PCA and PcoA are usually used as input to other statistical methods such as regression and machine learning or for visualization.

Few studies have shown the application of t-SNE to the microbiome data. (Xu et al., 2020) proposed a method to classify microbiome samples using t-SNE embeddings. They utilized the Aitchison distance for modifying the conditional probabilities in t-SNE to consider the compositional nature of microbiome data.

Other methods that are popular in microbiome data analysis are canonical correspondence analysis (CCA) (Ter Braak, 1986), non-metric multidimensional scaling (nMDS) (Clarke, 1993) and uniform manifold approximation and projection (UMAP) (McInnes et al., 2018). CCA is the most commonly used method for feature interpretation in microbiome data analysis. The reason for the wide usage of this method is its ability to incorporate sample metadata into the low-rank embeddings. nMDS is a non-linear method widely used in microbiome data analysis due to the possibility of incorporating an arbitrary dissimilarity measure. Like t-SNE, UMAP is mainly used for the visualisation and exploration of microbial community and diversity as well as identifying potential associations between microbial data and environmental

variables. Moreover, one advantage of UMAP is that it can handle non-linear relationships between microbiome features, which is beneficial considering the complex non-linear interactions in microbiomes. Moreover, it is worth mentioning that the classical practice in microbiome data analysis is to use a feature table as an input to dimensionality reduction methods which omits the important interactions between microbes, even though recently presenting the microbiome data as a network instead of a feature table (or OTU table) has been gaining some attention.

In terms of graph representation learning, it has not been commonly applied to analyzing microbiome dynamics. Most of these works utilize graph neural networks for predicting different associations, for example, between microbiome and drugs (Long et al., 2020). Therefore, to the best of our knowledge, this thesis is unique and unprecedented in its attempt at combining microbiome analysis and graph representation learning.

1.4 Thesis Overview

This thesis aims to enhance the analysis of microbiome data by developing methods that address the challenges discussed in Section 1.1, the complexity of multiple interactions of species presented in the microbiome data and finally, take into account the metastable behaviour of the microbiome compositions. We believe that it is crucial to represent the microbiome as a time-evolving graph to account for thousands of interactions between species and consider the metastable nature of the microbiome during its analysis. We propose to construct a series of correlation networks at different time points. In this case, nodes will represent species and edges will be determined as interactions between two species at the current time step. Constructing a correlation network on each time step has a major advantage. First of all, it can reveal temporal patterns in the community structure and help to identify species that are driving community dynamics. Second, it can help to determine the important biomarkers (species and interactions between species) that are affected by a temporal perturbation such as diet, disease, or medication.

Furthermore, unlike other methods that utilize the concept of networks, we believe that to gain a better understanding of the high-dimensional microbiome data, it is essential to reduce the data dimensionality in such a way that increases interpretability while simultaneously minimizing information loss. That is why, this

thesis proposes a new approach to the analysis of microbiome dynamics. Namely, instead of analysing the dynamic in the high-dimensional space, which is typical for microbiome data, we suggest first projecting it to a lower-dimensional space retaining the metastable behaviour and accounting for the complex interactions, we analyze its dynamical properties in this new space. In order to facilitate the finding of microbial biomarkers or interactions between microbes that are associated with transitions of the microbiome composition from one state to another, we complement our methods with interpretability approaches. We aim to address the following research questions:

1. **RQ1:** How can the microbiome metastability and a large number of interactions between species be accounted for in a method?

2. **RQ2:** How can data-driven dimension reduction techniques for dynamical systems be used to project the time-evolving graph constructed from microbiome data into a low-dimensional space while preserving its metastable properties?

3. **RQ3:** How effective is the use of deep learning technologies in learning the low-dimensional representation of time-evolving graphs while retaining the metastable behaviour of the microbiome?

4. **RQ4:** How can the identification of keystone species and interactions between them that play a key role in the transition of microbiome constitution from one state to another under various perturbations such as diet, disease, or antibiotic exposure be improved?

So far, we have discussed what the microbiome is and why it is important to analyze the microbiome, the challenges associated with microbiome data, and the current methods used for the analysis of microbiome dynamics. In Section 1.3, we have presented the difference between dimensionality reduction and graph representation learning. For each of these, we also reviewed state-of-the-art methods both in general and specific to microbiome data. The chapters 2, 3 and 4 will focus on the following analysis:

Exploring graph kernels and dynamical system theory as methods to analyse the human microbiome dynamics: In Chapter 2, we present an approach, which we call graphKKE and which is based on the spectral analysis of transfer operators, such as the Perron–Frobenius or Koopman operator, and graph kernels. We

explore the idea of analysing the microbiome dynamics by projecting it in a low-dimensional space so that there is no need to analyze the microbiome data in the high-dimensional space where the microbiome usually exists. We will show that the method is able to preserve the metastability properties in the low-dimensional space such as the number of metastable states and their location. The proposed method enables both the possibility of representing the microbiome data as a time-evolving graph and incorporating the energy landscape paradigm. Furthermore, Chapter 2 will present the microbiome and simulated datasets used in this thesis.

Deep learning for the analysis of the microbiome dynamic: Chapter 3 will investigate how deep learning technologies can be used to study the complex network of microbial species. We will utilize the state-of-the-art architecture in the sequential data analysis, the so-called transformer, and combine it with contrastive learning. The former allows us to learn the low-dimensional representation of the time-evolving graph while contrastive learning ensures that a new space contains the same dynamical properties as the high-dimensional space.

Extracting high-impacting structural graph patterns from the time-evolving graph: In Chapter 4, we address an important challenge in microbiome analysis which is feature importance. We will study how using two presented methods in the previous chapters, we can extract hubs and modules of the time-evolving graph that relate to each particular metastable state. From a biological point of view, we want to identify keystone species and interactions between them that play a central role, either as a driver or as a consequence, in the fluctuations of the microbiome composition under the influence of various factors such as diet, disease, or medication.

Chapter 2

A graph kernel-based approach for human microbiome analysis

The human body is a remarkable ecosystem where microbial life thrives, with approximately one out of every two cells being a microbial cell. This diverse community of bacteria, archaea, and viruses, collectively known as the microbiome, colonizes various parts of our body, including the skin, mouth, and gut, making up an astonishing 1.5 kg of symbiotic organisms. The presence of these microorganisms profoundly impacts both their host and the surrounding environment. Therefore, the influence of the microbiome on human health and disease has garnered significant attention in recent years. While efforts to understand the differences in microbiome profiles between healthy and ill individuals are underway, many studies have primarily focused on statistical constitution analysis, pairwise interactions, or static networks of species interactions. Unfortunately, such approaches often overlook more complex microbe-microbe and host-microbe interactions, which play pivotal roles in shaping the microbiome's dynamics and its impact on the host. Furthermore, as discussed in the previous chapter, the composition of the microbiome can be likened to a stability landscape, featuring a stable state representing a healthy microbiome and alternative states that emerge in response to various disturbances. In this thesis, we will refer to this phenomenon as metastable behavior.

The existing methods do not only ignore more complex interactions between species and temporal changes in the microbiome composition but also usually suffer from the curse of dimensionality, and overfitting due to the high dimensionality and complexity of the microbiome. Therefore, as the first step in the analysis of the microbiome, we propose modelling the microbiome as a set of correlation networks, a so-called time-evolving graph, constructed at each time step, which allows us to analyze the metastable behaviour and determine the species or interactions associated with the fluctuations of microbiome composition. To the best of our knowledge, we are the first who consider the metastable behaviour in graph-structured data. As an illustration of a time-evolving graph that lies in an energy landscape with two metastable states, consider the time-evolving microbiome interaction graph shown in Figure 2.1, where vertices represent the concentrations of bacteria species and edges pairwise associations between them. In this example, a disease can be thought of as a perturbation that displaces the microbiome composition from its *equilibrium (healthy) state*, which we call an *alternative state*. The consequence of this displacement is the reduction of the concentration in some vertices and the removal of corresponding edges. Given an evolution of the graphs (in this example, the evolution of the microbe interactions), we aim to analyze dynamics occurring in the graph

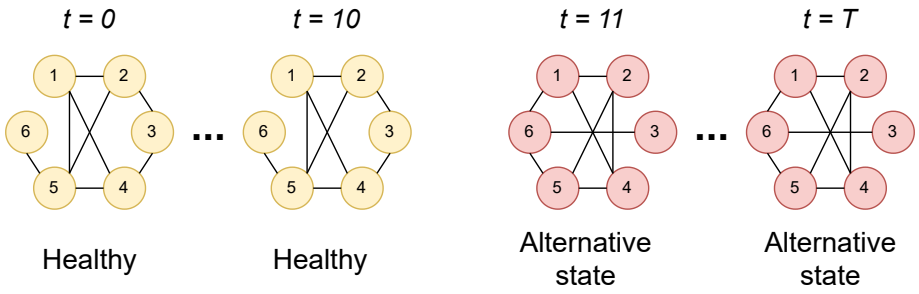


Figure 2.1: An example of a time-evolving graph of microbe interactions with two metastable states: *healthy* and *alternative state*. Two metastable states correspond to different structural graph features. For example, the stable state (healthy) is characterized by removed edges between the following node (species) pairs: $\{(1,6),(2,4),(3,6)\}$. The alternative state, in turn, is characterized by removed edges between the following node (species) pairs: $\{(1,5),(2,3),(3,4)\}$.

over time, namely, extracting the number of metastable states and their locations, substructures of a graph, which characterize the state space (e.g., the difference in the microbe interactions between the *healthy* or *stable state* and *alternative state*).

This proposed overlook on the microbiome data requires an approach that will be able to capture both temporal and topological patterns of time-evolving networks. To capture both, we propose learning the low-dimensional representation of the time-evolving graph which can then be used to analyze the metastable properties of the microbiome. In such a way, during learning of the low-dimensional representation, we can account for both temporal and topological patterns of the time-evolving graph.

In order to capture the temporal or dynamical properties of the time-evolving graph while learning the low-dimensional representation, we make use of the transfer operators theory. The approximation of transfer operators and their eigenfunctions has important applications in molecular dynamics, fluid dynamics, control theory, and many other areas. Transfer operators propagate probability densities or observables. Since these operators are infinite-dimensional, they are typically projected onto a space spanned by a set of predefined basis functions. Fortunately, the integrals required for this projection can be estimated from training data, resulting in methods such as extended dynamic mode decomposition. The finite-dimensional approximation of the transfer operator can be obtained by different kernel-based empirical estimates in two different ways: using explicitly the feature amp representation of the kernel and the second one is based on the kernel evaluations. This is where we propose to use graph kernels for both approximating the transfer operators

and extracting the topological information from the time-evolving graph.

To set the stage, in Section 2.1 we will first present the theory behind the transfer operators, how it can be used to learn the low-dimensional representation, and which data-driven methods exist to approximate the transfer operators. The following section 2.2 will introduce the concept of graph kernels, how they are used to measure the similarity of graph or the so-called *graph isomorphism*, and which main categories of graph kernels exist. In Section 2.3, our approach for learning the low dimensional representation of a time-evolving graph with metastability will be presented. Finally, in Section 2.4 and Section 2.5 we will discuss synthetic and real-world data used to evaluate our method and the main findings and results. However, before introducing the main components of our approach, we first define necessary notations and definitions that will be used throughout this thesis.

Definition 2.0.1. *A finite graph $G(V, E)$ is defined by*

- *a set of **nodes** (states, vertices) $V(G)$, and*
- *a set of **edges** (links, connections) $E(G) \subseteq V(G) \times V$ between nodes.*

If for two nodes $u, v \in V(G)$ there exists an edge, then we say that u and v are *neighbours*. In the graph theory, the **order** of a graph is its number of vertices $|V| = n$ and the **size** of a graph is its number of edges $|E|$. We will refer to $V(G)$ and $E(G)$ as sets of vertices and edges associated with graph G, respectively.

Depending on the nature of interactions in the system, a graph can be **directed** and **undirected**. For instance, a directed graph can be used to describe associations when one species influences another species but not the other way around. On the other hand, an undirected graph can represent cases where we do not know which species influences other species but we know that they have an association. In this case, we make no distinction between (u, v) and (v, u).

Another special case is formed by a simple graph having n vertices, with each vertex being adjacent to every other vertex. This graph is also known as a **complete graph**.

Definition 2.0.2. *The degree of a vertex, **def(v)** is the number of edges that are incident with the vertex.*

2.1 Transfer operators and microbiome dynamics

In this section, we will introduce the transfer operator theory and how it can be used to study microbiome dynamics. We start with defining a reproducing kernel Hilbert space, what the transfer operators are and which data-driven methods for the approximation of transfer operators exist.

2.1.1 Reproducing kernel Hilbert space

Definition 2.1.1. *Let $\mathbb{X}$ be as a set and $\mathbb{H}$ a space of functions $f : \mathbf{X} \to \mathbb{R}$. Then $\mathbb{H}$ is called a reproducing kernel Hilbert space (RKHS) with corresponding dot product $\langle \cdot, \cdot \rangle_{\mathbb{H}}$ and induced norm $\| f \|_{\mathbb{H}} = \langle f, f \rangle_{\mathbb{H}}^{1/2}$ if there is a function $k : \mathbb{X} \times \mathbb{X} \to \mathbb{R}$ such that*

1. *$\langle f, k(x, \cdot) \rangle_{\mathbb{H}} = f(x)$ for all $f \in \mathbb{H}$;*
2. *$\mathbb{H} = \overline{span\{k(x, \cdot) | x \in \mathbb{X}\}}$.*

The function k is called a reproducing kernel of $\mathbb{H}$. The first requirement implies that $\langle k(x, \cdot), k(x', \cdots) \rangle_{\mathbb{H}} = k(x, x')$ and is called *a reproducing property* of $\mathbb{H}$. Moreover, we can regard $k(x, \cdot)$ as a feature map $\phi(x)$ of x in $\mathbb{H}$ such that $k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathbb{H}}$. Thus, the reproducing kernel k is a kernel in a usual sense. A feature map ϕ exists if and only if k is a positive-semidefined function. For instance, one of the most commonly used kernels is the Gaussian RBF kernel with

$$k(x, y) = \exp \frac{- \| x - y \|_2^2}{2\sigma^2} \tag{2.1}$$

for $x, y \in \mathbb{X}$ and σ being a bandwidth parameter. In this Section, we will present a list of well-known graph kernels that are more suitable for graph-structured data. However, in the experiment section we will show that if we treat an adjacency matrix of a graph as a vector, it is possible to apply the Gaussian RBF kernel, even though the Gaussian RBF kernel will most likely omit the topological structure of the graph.

In order to evaluate a kernel k, we need to have the inner product between $\phi(x)$ and $\phi(x')$ in $\mathbb{H}$. However, in most applications, the kernel is normally not defined by an explicit representation of ϕ, but instead, each kernel implicitly defines a potentially infinite-dimensional mapping ϕ. This means that the kernel trick allows us to evaluate it directly without constructing ϕ.

2.1.2 Transfer operators

Transfer operators play a fundamental role in understanding the evolution of probability distributions and observables over time, enabling the investigation of the long-term behaviour of dynamical systems. These operators find wide-ranging applications across various scientific and engineering disciplines. For instance, they have proven invaluable in the analysis of molecular dynamics (Nuske et al., 2014; Schwantes and Pande, 2015), shedding light on the intricate motions of molecules and the underlying kinetics involved. Similarly, in the domain of fluid dynamics (Schmid, 2010), transfer operators help reveal complex flow patterns and provide insights into turbulence and stability phenomena.

The spectral properties of transfer operators are of paramount importance. They encapsulate key characteristics of the dynamical system under study. By analyzing the spectrum of these operators, one can discern essential information such as dominant modes of behaviour, stability, and long-term trends in the system. Moreover, eigenvalues and eigenvectors of transfer operators furnish vital clues about the system's metastable states, attracting regions, and the underlying connectivity between different states. These spectral properties are instrumental in understanding the system's overall behaviour and formulating effective low-dimensional representations, as will be demonstrated in the graphKKE approach. However, before presenting a novel method for the analysis of microbiome dynamics, the necessary definitions will be discussed.

Definition 2.1.2 (Time-homogeneous or stationary process)**.** *A stochastic process $X_{t\geq 0}$ is called a time-homogeneous, or stationary, if it holds for every $t \geq s \geq 0$ that the distribution of X_t conditional to $X_s = x$ only depends on x and (t-s).*

Definition 2.1.3 (Transition density)**.** *The transition density function $p_\tau : \mathbb{X} \times \mathbb{X} \to [0, \infty]$ of a time-homogeneous stochastic process $X_{t\geq 0}$ defined on the bounded state space $\mathbb{X} \in \mathbb{R}^d$ is expressed as*

$$\mathbb{P}[X_{t+\tau} \in \mathbb{A} | X_t = x] = \int_{\mathbb{A}} p_\tau(x, y) dy, \tag{2.2}$$

for every measurable set $\mathbb{A}$. In other words, $p_\tau(x, y)$ is the conditional probability density of $X_{t+\tau} = y$ given that $X_t = x$, for every measurable set $\mathbb{A}$.

Definition 2.1.4 (Transfer operator)**.** *Let p_t be a probability density, $f_t \in L^\infty(\mathbb{X})$ an*

observable of the system, and $u_t(x) = \pi(x)^{-1} p_t(x) \in L^1_\pi(\mathbb{X})$ *is a probability density with respect to the equilibrium density* π*.*

- *The Perron–Frobenius operator* $\mathcal{P}_\tau : L^1(\mathbb{X}) \to L^1(\mathbb{X})$ *is defined by*

$$\mathcal{P}_\tau p(x) = \int_{\mathbb{X}} p_\tau(y, x) p(y) dy. \tag{2.3}$$

- *The Koopman operator* $\mathcal{K}_\tau : L^\infty(\mathbb{X}) \to L^1(\mathbb{X})$ *is defined by*

$$\mathcal{K}_\tau f(x) = \int_{\mathbb{X}} p_\tau(x, y) f(y) dy = \mathbb{E}[f(X_{t+\tau}) | X_t = x]. \tag{2.4}$$

The time-homogeneity of the stochastic process $\{X_t\}_{t \geq 0}$ implies the *semigroup property* of the operator, i.e., $\mathcal{P}_{t+\tau} = \mathcal{P}_t \mathcal{P}_\tau$ and $\mathcal{K}_{t+\tau} = \mathcal{K}_t \mathcal{K}_\tau$ (Klus et al., 2018b). This property is crucial because it ensures that the evolution of observables or probability densities in a dynamical system follows consistent and well-behaved patterns. Particularly, it implied that these operators describe time-stationary Markovian dynamics. The Koopman operator describes the evolution of arbitrary observables while the Perron-Frobenous operator describes the evolution of densities.

Another aspect of understanding the statistical properties and stability of dynamical systems is equilibrium densities. The study of equilibrium densities involves finding the fixed points or invariant measures of the dynamical systems. These fixed points represent the states that do not change under the dynamics of the system. More formally, the equilibrium density can be defined as follows.

Definition 2.1.5. *A density* π *is called an invariant density or equilibrium density if* $\mathcal{P}_\tau \pi = \pi$*. That is, the equilibrium density* π *is an eigenfunction of the Perron-Frobenius operator* $\mathcal{P}_\tau$ *with corresponding eigenvalue 1.*

Then, the Perron-Frobenous operator with respect to the equilibrium density can be expressed as:

Definition 2.1.6. *The Perron–Frobenius operator with respect to the equilibrium density* $\mathcal{T}_\tau$ *is defined by*

$$\mathcal{T}_\tau u_t(x) = \int_{\mathbb{X}} \frac{\pi(y)}{\pi(x)} p_\tau(y, x) u_t(y) dy. \tag{2.5}$$

The operators $\mathcal{P}_\tau$ and $\mathcal{T}_\tau$ are often referred to as forward operators, while $\mathcal{K}_\tau$ is referred as backward operator. As said, the spectral properties of the transfer operator play an important role in the dynamical systems. Let $\mathcal{T}_\tau$ and $\mathcal{K}_\tau$ be the Perron-Frobenous operator and the Koopman operator, respectively. Then, we are particularly interested in computing eigenvalues $\lambda_l(\tau)$ and eigenfunctions φ_l of transfer operators, i.e.:

$$\mathcal{T}_\tau \varphi_l = \lambda_l(\tau)\varphi_l \tag{2.6}$$

for the Perron-Frobenous operator, or

$$\mathcal{K}_\tau \varphi_l = \lambda_l(\tau)\varphi_l \tag{2.7}$$

if we are interested in the Koopman operator. Given the eigenvalues and eigenfunctions of transfer operators, we can predict the evolution of the dynamical system. Furthermore, we assume that our dynamical system is reversible:

Definition 2.1.7 (Reversibility)**.** *A system is considered to be reversible if the detailed balance condition*

$$\pi(x)p_\tau(y|x) = \pi(y)p_\tau(x|y) \tag{2.8}$$

holds for all $x, y \in \mathbb{X}$.

In a reversible dynamical system, the transfer operator has a specific property known as adjointness. The adjoint of an operator is a concept similar to the transpose of a matrix but adapted to the space of probability distributions. Moreover, if the system is reversible, the operators' eigenvalues λ_l are real and the eigenfunctions φ_l constitute an orthogonal basis with respect to the corresponding scaler product. Consequently, the eigenvalues can then be sorted in descending order so that $1 = \lambda_1 > \lambda_2 \geq \lambda_3 \geq \dots$ (Klus et al., 2019a). For many applications, the dynamics in full-phase space are known to be reversible (Klus et al., 2018b). Yet, the reversibility of microbiome dynamics remains poorly understood.

The Perron-Frobenious operator maps densities p_t to densities $p_{t+\tau}$, while the Koopman operator maps an observable function f to its expected value function $\mathbb{E}[f(X_{t+\tau})|X+t=.]$. Furthermore, both the Koopman operator $\mathcal{K}_\tau$ and the Perron–Frobenius operator $\mathcal{P}_\tau$ are linear, infinite-dimensional operators, which are adjoint to each other and, therefore, it should not matter which one we choose to study the behaviour of the system. Although they are typically defined on the function spaces

L^1 and L^∞, we assume that the operators are well-defined on L^2 (for details, see Klus et al. (2016)).

The information about the long-term behaviour of the dynamical system is encoded in the spectral properties of these operators such as eigenvalues and eigenfunctions (Klus et al., 2019a). More precisely, eigenfunctions with eigenvalues close to 1 of both Koopman and Perron–Frobenius operators contain information about the locations of metastable states in the state space $\mathbb{X}$.

In order to compute the eigenfunctions of transfer operators numerically, the infinite-dimensional operators are projected onto a finite-dimensional space. Several methods have been proposed to approximate the transfer operator eigenvalue problem. We will discuss them further in this chapter.

2.1.3 Data-driven methods for approximation of transfer operators

This part of the thesis will describe different data-driven methods for approximating transfer operators. For a more elaborated description and the overarching connections between the various methods discussed, please refer to (**empty citation**).

Let $(x^{(i)}, y^{(i)})_{i=1}^m$ be data with $y^{(i)} = \Theta^\tau(x^{(i)})$ and Θ^τ is the flow map associated with the dynamical system. We start by giving the definitions of a covariance operator and a Gram matrix. Let k denote a positive define kernel on $\mathbb{X} \times \mathbb{X}$ which can be interpreted as the feature map $\psi(x)$ of x so that $k(x, x') = \langle \psi(x), \psi(x') \rangle_{\mathbb{H}}$.

Definition 2.1.8. *The covariance operator $C_{XX} : \mathbb{H} \to \mathbb{H}$ and cross-covariance operator $C_{YX} : \mathbb{H} \to \mathbb{H}$ are defined as*

$$C_{XX} = \int \psi(X) \times \psi(X) d\mu(X) \text{ and } C_{YX} = \int \mathcal{K}\psi(X) \times \psi(X) d\mu(X). \tag{2.9}$$

Since covariance operators cannot be analytically computed, we will consider empirical estimations from training data:

$$\hat{C}_{XX} = \frac{1}{n}\Phi\Phi^T = \frac{1}{n}\sum_{i=1}^n \phi(x_i) \times \phi(x_i), \tag{2.10}$$

$$\hat{C}_{YX} = \frac{1}{n}\Psi\Phi^T = \frac{1}{n}\sum_{i=1}^n \psi(y_i) \times \phi(x_i), \tag{2.11}$$

where $\Phi = [\phi(x_1), \ldots, \phi(x_n)]$ and $\Psi = [\psi(x_1), \ldots, \psi(x_n)]$ are feature matrices.

Another definition that will be used further in the section is the Gram matrix.

Definition 2.1.9. *The Gram matrix* $G_{XX} \in \mathbb{R}^{n\times n}$ *and time-lagged Gram matrix* $G_{YX} \in \mathbb{R}^{n\times n}$ *are defined as*

$$G_{XX} = \Phi^T\Phi = [k(x_i, x_j)]_{i,j=1}^n \text{ and } G_{YX} = \Psi^T\Phi = [k(y_i, x_j)]_{i=1}^n.$$

Time-lagged independent component analysis

The time-lagged independent component analysis (TICA) is a widely used dimension reduction method for time-series data. The method was first introduced in (Molgedey and Schuster, 1994) as a solution to the blind source separation problem. In dynamical systems, TICA is usually used as a preprocessing step to reduce the size of the state space by projecting the dynamics onto the main coordinates. Let us assume the dynamical system is reversible, then the TICA coordinates are eigenfunctions of $\mathcal{K}_\tau$ and $\mathcal{P}_\tau$ projected onto the space spanned by linear basis function. Let C_{XX} and C_{YX} be covariance and cross-covariance matrices defined in Eq. 2.9. Solving the following eigenvalue problem, we can obtain the time-lagged independent components

$$C_{XY}\varphi_l = \lambda_l C_{XX}\varphi_l \tag{2.12}$$

or

$$C_{XX}^+ C_{XY}\varphi_l = \lambda_l\varphi_l, \tag{2.13}$$

where C_{XX}^+ denotes the Moor-Penrose pseudo-inverse of C_{XX}.

The time-lagged independent components refer to statistically independent patterns or features that are extracted from time-series data.

Dynamic mode decomposition

Dynamical mode decomposition (DMD) is a data-driven method for extracting spatial and temporal patterns from multi-dimensional time series data and found its application in various fields such as mechanics, robotics, and neuroscience (Wu et al., 2021). Dynamical mode decomposition was developed as a tool to identify coherent structures in fluid flows (Wu et al., 2021). The basic idea behind DMD is to decompose a given time-series data into a set of spatial patterns (modes) that evolve dynamically over time. These modes capture the dominant coherent structures and oscillatory behaviour present in the system

In DMD we assume that there exists a linear operator M such that $y_i = Mx_i$. Since the underlying dynamical system is in general nonlinear, this equation cannot be fulfilled exactly and we want to compute the matrix M in such a way that the Frobenius norm of the deviation is minimized:

$$min \parallel Y - M_{DMD}X \parallel_F . \tag{2.14}$$

The solution to this minimization problem is given:

$$M_{DMD} = YX^+ = (YX^T)(XX^T)^+ = C_{YX}^T C_{XX}^+ = M_{TICA}^T. \tag{2.15}$$

The eigenvalues and eigenvectors of M_{DMD} are called DMD eigenvalues and modes, respectively. In order to obtain DMD eigenvalues and modes, we need to solve the following equation:

$$M_{DMD}\varphi_l = \lambda_l \varphi_l. \tag{2.16}$$

Variational approach of conformation dynamics

The variational approach of conformation dynamics (VAC) is a generalization of the frequently used Markov state modelling framework that allows arbitrary basis functions. In addition to data, VAC requires a set of basis functions ψ, often called *dictionary*. The variational approach computes eigenfunction of $\mathcal{T}_\tau$ or $\mathcal{K}_\tau$. Using the definition of the covariance and cross-covariance matrices in Eq. 2.9, we define a VAC matrix as $M_{VAC} = C_{XX}^+ C_{XY}$. The matrix M_{VAC} can be regarded as a finite-dimensional approximation of $\mathcal{K}_\tau$. Eigenfunctions of the Koopman operator can be then approximated by the eigenvectors of the matrix M_{VAC}. Let ξ_l be an eigenvector of M_{VAC}:

$$M_{VAC}\varphi_l = \lambda_l \varphi_l \tag{2.17}$$

and $\phi_l(x) = \varphi_l^* \phi(x)$, where * denoted the conjugate transpose.

Extended dynamic mode decomposition

Extended dynamic mode decompositions (EDMD) is a generalization of DMD and can be used to compute finite-dimensional approximations of the Koopman operator, its eigenvalues, eigenfunctions, and eigenmodes (Williams et al., 2015). EDMD can also be used to approximate eigenfunctions of the Perron-Frobenius operator with

respect to the density underlying the data points. Moreover, EDMD does not require a dynamical system to be reversible, unlike VAC. With the notation of the covariance and cross-covariance above, the minimization problem can be written as follows

$$min \parallel \Psi_Y - M_{EDMD}\Psi_X \parallel_F . \tag{2.18}$$

The solution is then given by:

$$M_{EDMD} = \Psi_y \Psi_x^+ = (\Psi_Y \Phi_X^T)(\Psi_X \Psi_X^T)^+ = C_{YX}^T C_{XX}^+ = M_{VAC}^T. \tag{2.19}$$

EDMD aims at finding a linear relationship between the transformed data matrices Ψ_X and Ψ_Y instead of assuming a linear relationship between the data matrices X and Y.

Kernel-based discretization of eigenvalue problem

Unlike previous methods that are based on approximating the eigenvalues problem with covariance matrices, (Klus et al., 2018a) propose using kernel evaluations of training data for the numerical solution of the transfer operator eigenvalue problem. In fact, all previous methods discussed here can be regarded as special cases of this approach. One of the main advantages of this approach is instead of explicitly constructing high-dimensional feature space, kernels can be defined implicitly.

We first assume that the Perron-Frobenius operator and the Koopman operator are defined on $L^2_{\mathbb{X}}$. Kernel transfer operators follow then from the assumption that densities and observables in $L^2_{\mathbb{X}}$ can be represented as elements of the RKHS $\mathbb{H}$ (for more details see (Klus et al., 2018a)). Let us consider the original transfer operator eigenvalue problem $\mathcal{K}\varphi = \lambda\varphi$. Then, we aim at finding functions $\tilde{\varphi} \in \mathbb{H}$ such that:

$$\mathcal{K}\mathcal{E}_k\tilde{\varphi} = \lambda\mathcal{E}_k\tilde{\varphi}, \tag{2.20}$$

where $\mathcal{E}_k$ is the embedding operator $\mathcal{E}_k : L_2(\mu) \to L_2(\mu)$. In order to obtain an eigenfunction we set $\varphi = \mathcal{E}_k\tilde{\varphi}$. After a set of operations, we have an approximation of the transfer operator eigenvalue problem that can be solved numerically:

$$G_{YX}\tilde{\varphi_X} = \lambda G_{XX}\tilde{\varphi_X}, \tag{2.21}$$

where $\tilde{\varphi}_X = [\tilde{\varphi}(x_1), \ldots, \tilde{\varphi}(x_n)]$.

2.2 Graph kernels for microbiome network

2.2.1 What is graph kernel?

Graph kernels have become an established and widely-used machine learning technique for solving various tasks on graphs. Due to the abundance of graph-structure data, particularly in chemo- and bioinformatics domains, and the empirical success of kernel-based methods for learning from such data, many works in this area exist. Graph kernels are a family of algorithms used to compare and measure the similarity between graphs. Moreover, transforming graphs into feature vectors with graph kernels allows the use of machine learning methods on graph-structured data. Graph comparison is a fundamental problem with numerous applications in many different domains (Nikolentzos et al., 2021). Two graphs with identical structures are called *isomorphic*. However, the problem of graph isomorphism is very computationally demanding requiring a lot of computational resources, especially in the case of large graphs. Graph kernels try to tackle this issue in the most efficient way while also capturing the topological patterns in the graph. Intuitively, graph kernels can be considered as a bridge between graph-structure data and machine learning which, in most cases, are dedicated to work on vector-structure data and are not able to capture the topological complex interactions graphs. Another reason for the empirical success of graph kernels is that they allow the large family of kernels to work directly on graphs (Nikolentzos et al., 2021).

Over the past 20 years, many graph kernels have been proposed that can broadly be divided into two groups: (1) those that compare nodes in a graph and (2) those that compare graphs. Moreover, there exist many different criteria that can be used to divide graph kernels into different groups. For example, graph kernels can be divided into groups based on which structural aspects of graphs, such as walks, subtrees, paths, or subgraphs, they use to measure similarity. Another option is to group graph kernels based on their ability to handle unlabeled graphs, node-labelled graphs, or edge-labelled graphs. It is worth noting that even state-of-the-art graph neural networks follow a neighbourhood aggregation scheme similar to many graph kernels, and they can be combined in the same framework (Feng et al., 2022; Morris et al., 2019). For further reading, we refer to overviews of graph kernels provided in (Nikolentzos et al., 2021; Kriege et al., 2020).

In the context of the microbiome, graph kernels can be used to analyze the

relationships between different species. As we discussed in Chapter 1, it is very important to represent the microbiome as a time-evolving graph which makes graph kernels a logical choice to study the microbiome structural patterns and relationships between different microorganisms. Also, to understand the difference between the healthy state and the alternative state of the microbiome composition, we need to measure the similarity between graphs at different time steps.

We start the section by giving the formal definition of a kernel function. Then, we will give an overview of the most popular graph kernels that are categorized based on which structural components of the graph they operate on. In this thesis for our method, we will use a neighbourhood aggregation graph kernel, the 1-dimensional Weisfeiler–Lehman kernel since it is proven to be the most computationally efficient. However, one can potentially use other graph kernels, which can be tailored to specific applications and graph types.

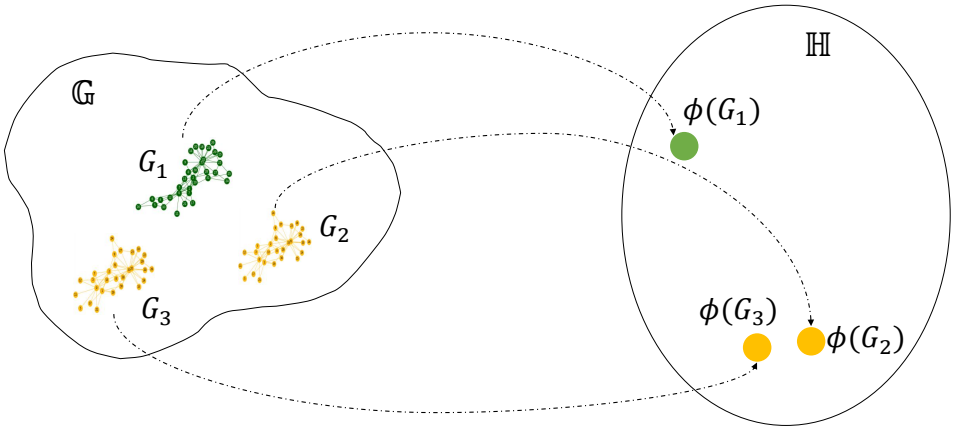


Figure 2.2: Illustration of mapping graphs into a Hilbert space $\mathbb{H}$ with a feature map ϕ. After graphs are mapped into the Hilbert space, a kernel on a space of graphs $\mathbb{G}$ can be defined as an inner product $k(G_1, G_2) = \langle \phi(G_1), \phi(G_2) \rangle$.

2.2.2 State-of-the-art graph kernels

In Section 2.1 we have given a definition of positive semidefined kernel, Gram matrix, and Hilbert space. These definitions are commonly used not only in the transfer operate domain but also in graph kernels-related fields. Kernel-based methods are machine learning algorithms that operate on input data after they have been mapped into an implicit feature space using a kernel function. The main advantage of kernel

methods is that they can be applied to very general types of data such as graphs or texts as long as we can find a mapping $\phi : \mathbb{X} \to \mathbb{H}$, where $\mathbb{H}$ is an RKHS. The illustration of mapping graphs into an RKHS is demonstrated in Figure 2.2. However, the kernel function is normally not defined by an explicit representation of ϕ, but instead, each kernel implicitly defines a potentially infinite-dimensional mapping ϕ.

All graph kernels can be divided into categories based on how they measure the similarity of graphs, whether they use node and edge label information, and on which structural features of the graph they are based. Since in the microbiome domain graphs can be constructed with different properties such as directed or undirected, unlabeled or node-labels/edge-labeled, we will describe each category and present some examples of graph kernels for each category.

Kernels based on subgraphs patterns

Graph kernels that are based on subgraphs patterns view graphs as bags of vertices and edges by ignoring global structure. These kernels work by counting the number of times a particular subgraph pattern appears in each graph and then computing a similarity score based on the frequency of occurrence of these subgraph patterns. For example, the *vertex histogram kernel* or *vertex label kernel* compares graphs only at the level of similarity between all pairs of vertex labels from two different graphs:

$$k(G, G') = \langle f_G, f_{G'} \rangle, \tag{2.22}$$

where $f_G = (f_G^1, \ldots, f_G^d)$ such that $f_G^i =\mid \{v \in V(G) : l(v) = i\} \mid$, $\forall i \in \Sigma$ and Σ being a set of node labels with $d =\mid \Sigma \mid$, $l : \mathcal{V} \to \Sigma$. Similar to the vertex histogram kernel, the *edge histogram kernel* is based on counting the number of occurrences of each label in an edge set of the graph. Unlike the vertex histogram kernel, the edge histogram kernel is able to capture the structural information of the graph. These two graph kernels are considered to be simple.

A downside of vertex and edge label kernels is that they ignore the structural properties of graphs and are completely uninformative in the case of node labels or edge labels being absent. Instead of considering a graph as a bag of vertices and edges, we can see them as bags of subgraphs patterns. Motivated by the graph reconstruction conjecture which states that a graph is determined uniquely by its subgraphs, (Shervashidze et al., 2009) proposed a *graphlet kernel*. It decomposes graphs into graphlets, where graphlets are small subgraphs with $k \in \{3, 4, 5\}$ vertices

and counts matching graphlets in the graphs:

$$k(G, G') = f_G^T f_{G'},$$

where f_G and $f_{G'}$ are vectors of the occurrence of each graphlet of size k. From the above definition, the graphlet kernel is computed using explicitly defined feature maps (Nikolentzos et al., 2021). One of the limitations of the graphlet kernel is that it does not work with node labels or edge labels. Moreover, the graphlet kernel can be computationally expensive, especially for large graphs. *Subgraph matching kernel* (Kriege and Mutzel, 2012) is another example from this category which counts the number of matching between subgraphs of bounded size in two graphs. This graph kernel can be applied to graphs that contain node labels, edge labels, node attributes or edge attributes.

Kernels based on paths and walks

In order to tackle a challenge of the subgraph pattern kernels which is the choice of a set of patterns or a subgraph size, there has been proposed an alternative of comparing the sequences of vertex or edge attributes that are encountered through traversals through graphs. One of these kernels is a shortest-path kernel. The idea of kernels based on shortest paths is to compare the attributes and lengths of the shortest paths between all pairs of vertices in two graphs.

Definition 2.2.1 (Shortest path)**.** *A shortest path from vertex v to vertex u of a graph G is a path from v to u such that there exists no other path between these two vertices with a smaller length.*

The first step of the shortest-path kernel is to transform the graphs into shortest-path graphs. The shortest-path graphs have the same set of nodes as an original graph but a different set of edges. Formally, given two graphs G and G' with label function $l : V(G) \cup V(G') \to \Sigma$ and let $d(u, v)$ be the shortest path distance between the vertices u and v in the same graph. Then, the kernel is defined as

$$k(G, G') = \sum_{(u,v) \in V(G)^2, u \neq v} \sum_{(u_1,v_1) \in V(G')^2, u_1 \neq v_1} k((u, v), (u_1, v_1)), \tag{2.23}$$

where

$$k((u,v),(u_1,v_1)) = k_L(l(u),l(u_1)) \cdot k_L(l(v),l(v_1)) \cdot k_D(d(u,v),d(u_1,v_1). \quad (2.24)$$

k_L and k_D are a kernel for comparing vertex labels and shortest-path distance, respectively.

Another example of graph kernels that travel through the graph is kernels that are based on random walks. The first graph kernels based on random walks were proposed by (Kashima et al., 2003; Gärtner et al., 2003b). The idea behind these graph kernels is that they count the number of walks that two graphs have in common. (Kashima et al., 2003) followed a probabilistic view of kernels and based his method on so-called marginalized kernels. The feature space of the kernel consists of all possible label sequences produced by random walks. In this version of the random walk kernel, the length of the walks is unbounded, which leads to the feature space being of infinite dimension. The computation of this random walk kernel is based on finding the stationary state of a discrete-time linear system. Graph kernels based on random walks, however, suffer from so-called *tottering*, which means that random walks may visit the same vertex several times. These repeated consecutive vertices do not provide useful information and may even bring unnecessary noise into similarity measures. That is why the marginalized graph kernel was extended to avoid tottering by replacing the underlying first-order Markov random walk model with a second-order Markov random walk model (Kriege, 2015).

Unlike the method discussed above, (Gärtner et al., 2003b) introduced a method based on the direct product graph of two labelled input graphs, which is defined as

Definition 2.2.2. *Direct product graph For two labeled graphs G and G', the direct product graph is denoted by $G \times G' = (\mathcal{V}, \mathcal{E})$, where*
$\mathcal{V} = \{(v,v') \in V \times V' \mid l(v) = l(v')\}$
$\mathcal{E} = \{(u,u'),(v,v') \in \mathcal{V} \mid (u,v) \in E \wedge (u',v') \in E' \wedge l(u,v) = l(u',v')\}$.

Then, the direct product kernel is

$$K_{RW}(G,G') = \sum_{i,j=1}^{|\mathcal{V}|} \Big[\sum_{l=0}^{\infty} \lambda_l A_{\times}^l \Big]_{ij}, \quad (2.25)$$

where $A_{\times}$ is the adjacency matrix of $G \times G'$ and λ is a sequence of weights such that the above sum converges. (Vishwanathan et al., 2010) proposed a generalization

framework for the random walk. Given an edge kernel k_E on attributes from the set $\mathcal{A}$. let $\psi : \mathcal{A} \to \mathbb{H}$ be a feature map. For an attributed graph G, the feature matrix $\Psi(G)$ is defined as $[\Psi(G)]_{ij} = \psi(v_i, v_j)$ if $(v_i, v_j) \in E(G)$ and 0 otherwise. The proposed kernel is then defined as follows.

$$K_{RW}(G, G') = \sum_{l=0}^{\infty} \mu_l \mathbf{q}_x^T \mathbf{W}_x^l \mathbf{p}_x \tag{2.26}$$

with $\mathbf{q}_x$, $\mathbf{p}_x$ being stopping and initial probability distribution, respectively. μ_l are coefficients such that the sum converges and $\mathbf{W}_x = \Psi(G) \otimes \Psi(G')$ is a weight matrix of the direct product graph $G \times G'$.

All above mentioned random walk kernels allow taking walks with unbounded length, which leads to infinite dimensional feature space. Several graph kernels based on random walks were proposed that use a certain length of walks. However, these kernels are application-related such as protein function (Borgwardt et al., 2005) or image classification (Harchaoui and Bach, 2007).

Neighborhood aggregation kernels

Neighbourhood aggregation approaches work by assigning an attribute to each vertex based on a summary of the local structure of its neighbourhood. For each vertex, the attributes of its neighbours are aggregated to compute a new attribute for the target vertex. The idea is that if two vertices have the same local structure, then their new attributes will match. One of the most popular and efficient graph kernels in this class is the Weisfeiler-Lehman kernel (WL) (Shervashidze et al., 2011). This graph kernel has many variations such as the Weisfeiler-Lehman edge, the Weisfeiler-Lehman shortest-path kernel or generalised Weisfeiler-Lehman kernel (Schulz et al., 2022).

After more than one decade, the WL kernels are still among the most prevalent graph kernels due to their predictive performance and time complexity. Therefore, we will use the original WL kernel in our method and discuss it in this section. Let G and $\widehat{G}$ be graphs and $l^{(0)}$ be a set of unique original vertex labels of G and $\widehat{G}$. The key idea of this kernel is to augment each vertex label by the sorted set of neighbouring vertex labels, and then compress the augmented label into some new label using a hash function f. That is, at each iteration $h = 1, \ldots$ the 1-dimensional

Weisfeiler–Lehman kernel computes a new set of vertex labels $l^{(h)}$ such that

$$l_v^{(h)} = f\Big(l_v^{(h-1)} + (l_{u_0}^{(h-1)} + ... + l_{u_k}^{(h-1)})\Big), \{u_0, ..., u_k\} \in \text{sorted}(\mathcal{N}(v)),$$

$\forall v \in V(G) \cup V(\widehat{G})$ and where the symbol "+" denotes the concatenation of strings, $\mathcal{N}(v)$ the set of neighbours of a vertex v, and sorted($\mathcal{N}(v)$) means that vertex labels need to be sorted before concatenation. The hash function f is chosen in such a way that $f(l^{(h)}(v)) = f(l^{(h)}(v'))$ if and only if $l^{(h)}(v) = l^{(h)}(v')$, $v, v' \in V(G) \cup V(\widehat{G})$. The next step is to compute a feature vector for each graph G and $\widehat{G}$ at each iteration h:

$$\varphi^{(h)}(G) = (C^{(h)}(G, l_0^{(h)}), ..., C^{(h)}(G, l_{|l^{(h)}|}^{(h)})),$$

where $l^{(h)} = \{l_0^{(h)}, l_1^{(h)}, \ldots, l_{|l^{(h)}|}^{(h)}\}$ denotes the set of compressed vertex labels at iteration h and $C^{(h)}(G, l_i^{(h)})$ is the number of occurrences of a label $l_i^{(h)}$ in the graph G at iteration h.

Finally, the Weisfeiler–Lehman kernel for two graphs G and $\widehat{G}$ is defined as:

$$k(G, \widehat{G}) = \langle \varphi^{(0)}(G), \varphi^{(0)}(\widehat{G}) \rangle + ... + \langle \varphi^{(h)}(G), \varphi^{(h)}(\widehat{G}) \rangle.$$

We chose the WL kernel because it outperformed other kernels in terms of runtime in our experiments. According to (Shervashidze et al., 2011), the WL subtree kernel on a pair of graphs can be computed in time $O(hm)$, where h is the number of iterations and m the number of edges, whereas the random walk kernel (Gärtner et al., 2003a) on a pair of graphs has the runtime complexity $O(n^6)$, where n is the number of nodes. Moreover, it is also competitive in terms of accuracy with state-of-the-art kernels. But, as mentioned above, one could use other graph kernels as well. The optimal choice depends strongly on the dataset.

In many real-world applications, the microbiome, in particular, nodes in a graph may have continuous labels such as numerical values or vectors. However, as it is clear from the graph kernels described above, they are mainly designed to work with discrete labels such as categorical labels or binary values. To address this challenge, researchers have developed various techniques for incorporating continuous node labels into graph kernels. Some of these techniques are to transform the continuous node labels into discrete labels before applying a traditional graph kernel or one can use kernel functions that are specifically designed to handle continuous data, such

as the Gaussian kernel or the Laplacian kernel. In one of our experiments in this thesis, we will apply the Gaussian kernel directly to a time-evolving graph. Further, we will discuss graph kernels that can work with continuous node attributes in more detail.

Graph kernels with continuous labels

Neighbourhood aggregation kernels, kernels based on pattern counting or kernel functions based on walks and paths measure similarity between graphs by considering only discrete node or edge attributes. However, many real-world graphs contain continuous real-valued node attributes. Unfortunately, designing graph kernels for graphs with continuous node labels is a much less studied problem than graph kernels for unlabeled graphs or for graphs with discrete node labels. However, recently this problem started drawing more and more attention. As mentioned above there are two possible approaches to handling graphs with continuous node labels: 1) graph kernels that are specifically developed to handle this type of graphs or 2) using discretization of node labels before employing existing graph kernels that operate on graphs with discrete node labels. An example of an approach from the second category is a hash graph kernel proposed by (Morris et al., 2016). The authors introduce a generic method that iteratively hashes continuous labels to discrete labels.

The examples of graph kernels designed to work on graphs with continuous node labels are subgraph matching kernel (Kriege and Mutzel, 2012), which has been mentioned earlier in this section. The GraphHopper kernel (Feragen et al., 2013) can operate on both discrete node labels and continuous attributes. The idea of the GraphHopper is similar to the shortest path kernel where it compares the shortest paths between node pairs from the two graphs but using a different path kernel. Further examples of graph kernels that can handle graphs with continuous labels include Propogation Kernel (Neumann et al., 2016), Graph Invariant kernel (Orsini et al., 2015), Multiscale Laplacian graph kernel (Kondor and Pan, 2016).

2.3 Novel unsupervised approach for time-evolving graph embedding based on graph kernel and transfer operators

Microbiome dynamics play a crucial role in understanding changes occurring in the microbiome constitution that undergoes transitions from the health state to an alternative state. However, the high dimensionality of microbiome data makes the analysis of microbiome dynamics challenging. Therefore, in this section, we will present a dimensionality reduction method for time-evolving graphs which is based on transfer operator theory and graph kernels. Graph kernels are used to extract the structural properties of time-snapshot graphs such as complex interactions between species in the microbiome and approximating the transfer operator such as the Koopman operator or Perron-Frobenious will allow us to learn the dynamics occurring in a time-evolving graph. In our experiment, we demonstrate how to use our method on real-world microbiome data. In order to evaluate our method, we construct synthetic data that will represent a time-evolving graph with metastable behaviour. We begin the section by formulating the problem. An illustration of the proposed method is shown in Figure 2.3.

2.3.1 Problem Formulation

Given a time-evolving graph $\mathcal{G}$ as a sequence of T graphs $\mathcal{G} = (G_0, \ldots, G_{T-1})$ at the consecutive time points $\{0, \ldots, T-1\}$ for some $T \in \mathbb{N}$. We call G_t *a time-snapshot graph* of $\mathbb{G}$ at time t. We focus, in particular, on the metastability of the time-evolving graph, that is, the property of being stable for a long time, and occasionally undergoing critical transitions from one state to another state with a significant change in the structure of the time-evolving graph.

Definition 2.3.1. *The time-evolving graph $\mathcal{G}$ exhibits metastable behavior if $\mathcal{G}$ can be partitioned into s subsets $\mathcal{G} = \mathcal{G}_0 \cup \cdots \cup \mathcal{G}_{s-1}$ for some $s \ll T$ such that for each time point $t \in \{0, \ldots, T-1\}$*

$$P(G_{t+1} \in \mathcal{G}_i \mid G_t \in \mathcal{G}_j) \ll 1, \text{ if } i \neq j$$

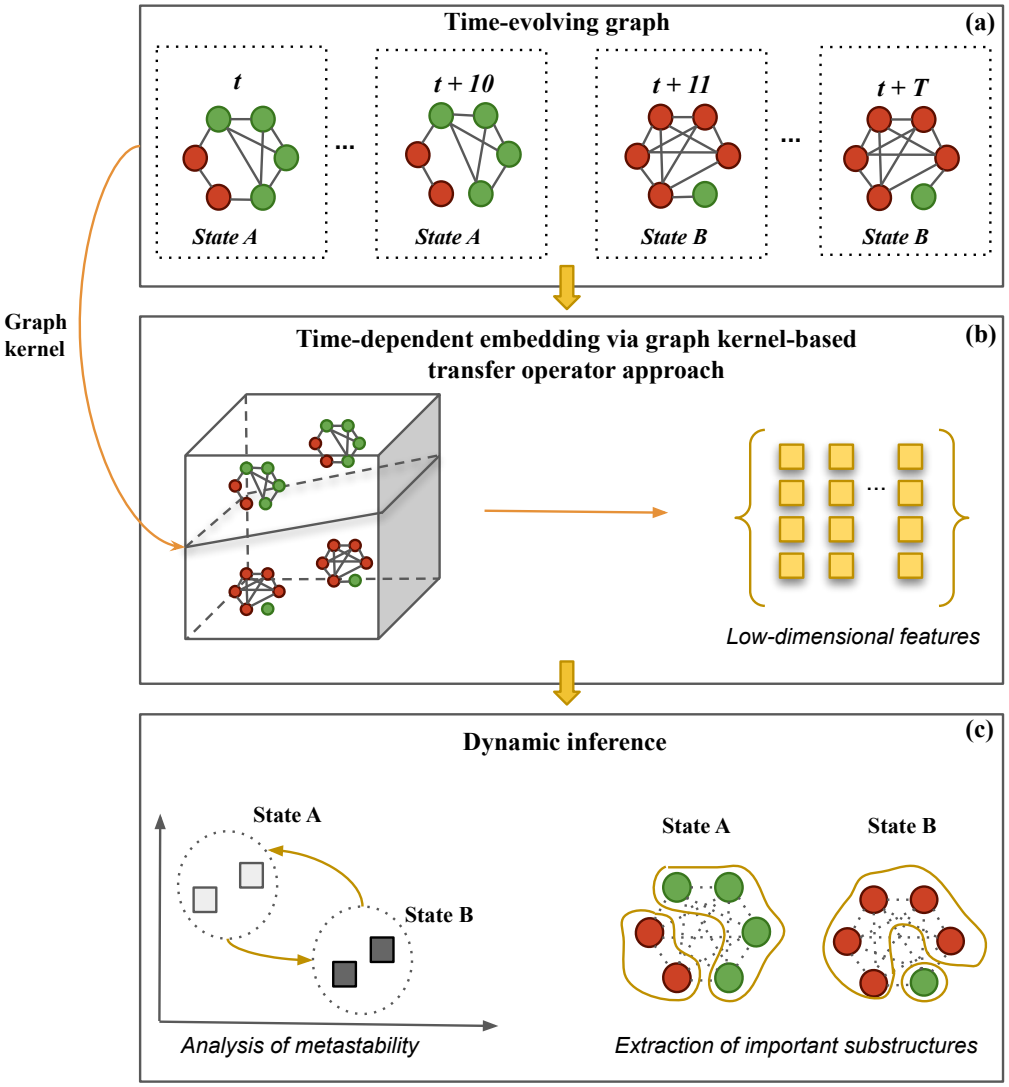


Figure 2.3: The illustration of the proposed method. **(a)** Constructing a time-evolving graph from the microbiome. **(b)** Approximating spectral properties of transfer operators using graph kernels, where $k(\cdot, \cdot)$ is a graph kernel and $\mathcal{K}_k$ is the Koopman operator. **(c)** In the learned embedding space it is possible to analyze the microbiome dynamics such as detecting metastable states and determining distinct substructures.

and

$$P(G_{t+1} \in \mathcal{G}_i \mid G_t \in \mathcal{G}_j) \approx 1, \text{ if } i = j.$$

We call $\mathcal{G}_0, \ldots, \mathcal{G}_{s-1}$ metastable states of the time-evolving graph $\mathcal{G}$ and each $G_t, t = 0, \ldots, T-1$, belongs to exactly one of the states $\mathcal{G}_i$. In most cases, each state $\mathcal{G}_i$ has to be characterized by a certain pattern in the graph structure such as node or edge clusters. We define our problem as follows: *Given a time-evolving graph* $\mathcal{G} = (G_0, \ldots, G_{T-1})$ *with assumed metastable behaviour, we aim to represent each time-snapshot* G_t *as a vector in a low-dimensional space* $\mathbb{R}^m$*, where* m *is a number of embedding dimensions, retaining the metastable behaviour of* $\mathcal{G}$*.*

Commonly, the number of embedding dimensions m is a hyperparameter that has to be tuned in order to obtain a good performance For our approach we will however show that the number of embedding dimensions m is defined by the number of states s, which eliminates the need to optimize this hyperparameter.

2.3.2 GraphKKE: Graph kernel Koopman Embedding for Human Microbiome Analysis

In order to address the problem defined above, we need to account for both structural information and temporal changes in a time-evolving graph. In the case of the microbiome, the structural information corresponds to the complex interactions between species that are impacted by the transition from one state to another, for example, from the period of antibiotic exposure to the period of recovery. The temporal changes, in turn, represent the transition from one state to another and respective changes in the structure of the time-evolving graph. In our approach, we propose to use transfer operator theory to capture the temporal changes and graph kernels to understand the structural patterns of the time-evolving graph. Our approach is inspired by kernel EDMD discussed in Section 2.1.3.

As it was discussed in Section 2.1, transfer operators, both the Koopman operator and Perron–Frobenious are infinite-dimensional, therefore it is necessary to find a suitable finite-dimensional subspace. It was shown that the initial eigenvalue problem on L^2 can be approximated by an eigenvalue problem defined on the reproducing kernel Hilbert space $\mathbb{H}$ utilizing only kernel evaluations. Assume we have measurement data given by a time-evolving graph $\mathcal{G} = (G_0, ..., G_{T-1})$, where each G_t is a time-snapshot graph of $\mathcal{G}$ at time point t and $\widehat{\mathcal{G}}$ is a set of graphs mapped

forward for a time lag τ, that is, $\widehat{G}_t = G_{t+\tau}$. It was shown for vector-structure data in Section 2.1.3 that in order to find eigenfunctions of transfer operators, we need to solve auxiliary matrix eigenvalue problems which are formulated as follows.

$$K_{\mathcal{G}\mathcal{G}}^{-1} K_{\widehat{\mathcal{G}}\mathcal{G}} \tilde{\varphi} = \lambda \tilde{\varphi} \tag{2.27}$$

and

$$K_{\mathcal{G}\mathcal{G}}^{-1} K_{\mathcal{G}\widehat{\mathcal{G}}} \tilde{\varphi} = \lambda \tilde{\varphi}, \tag{2.28}$$

where $[K_{\mathcal{G}\mathcal{G}}]_{ij} = k(G_i, G_j)$, $[K_{\widehat{\mathcal{G}}\mathcal{G}}]_{ij} = k(\widehat{G}_i, G_j)$ denote Gram matrices, $k(\cdot, \cdot)$ is a graph kernel, and $K_{\mathcal{G}\widehat{\mathcal{G}}} = K_{\widehat{\mathcal{G}}\mathcal{G}}^{\top}$. The equations (2.27) and (2.28) approximate the Koopman operator and Perron–Frobenius operator, respectively. This eigenvalue problem is closely related to kernel canonical correlation analysis (kernel CCA), see (Klus et al., 2019b). Kernel CCA computes eigenfunctions of the forward-backward dynamics to identify so-called coherent sets. Coherent sets are a generalization of metastable sets and are regions of the state space that are not distorted over a certain time interval.

Additionally, in order to evaluate the eigenfunctions of these operators at a given graph, we set

$$\varphi = \Psi \tilde{\varphi},$$

if $\tilde{\varphi}$ is the solution of the eigenvalue problem (2.27).

Otherwise, if $\tilde{\varphi}$ is the solution of the eigenvalue problem (2.28), we set

$$\varphi = \Psi K_{\mathbb{G}\mathbb{G}}^{-1} \tilde{\varphi},$$

where $\Psi = [k(\cdot, G_0), \ldots, k(\cdot, G_{T-1})]$ is called a feature matrix.

We assume that $K_{\mathcal{G}\mathcal{G}}$ is non-singular or otherwise we replace the inverse by its regularized version $(K_{\mathcal{G}\mathcal{G}} + \eta I)^{-1}$, where $\eta \geq 0$ is a ridge parameter. This regularization is known as Tikhonov regularization.

Furthermore, if $k(\cdot, \cdot)$ is a graph kernel, then we apply the following normalization:

$$k_{norm}(G_i, G_j) = \frac{k(G_i, G_j)}{\sqrt{k(G_i, G_i)\, k(G_j, G_j)}},$$

for all $i, j = 0, \ldots, T-1$. The same normalization is applied to both $\mathcal{G}$ and $\widehat{\mathcal{G}}$.

In our experiments, we will use a neighbourhood aggregation kernel — the

Weisfeiler–Lehman (WL) kernel — for graphs with discrete vertex labels discussed in Section 2.2. However, depending on the type of node labels and edge labels, one has to choose the right graph kernel that is able to capture the necessary information present in the time-evolving graph.

The number of states s in the time-evolving graph $\mathcal{G}$ is determined by the number of dominant eigenvalues close to 1. That is, if we have s dominant eigenvalues close to 1, then the time-evolving graph can be divided into s subsets $\mathcal{G} = \mathcal{G}_0 \cup \cdots \cup \mathcal{G}_{s-1}$. Moreover, all information about the long-term behaviour of the time-evolving graph $\mathcal{G}$ is contained within the eigenfunctions associated with s dominant eigenvalues close to 1. All things considered, the dominant eigenvalues can be used to determine the number of states s in the data and the dimension of a new low-dimensional space. The eigenfunctions associated with the dominant eigenvalues close to 1 are considered as a low-dimensional representation of the time-evolving graph $\mathcal{G}$ (Melnyk et al., 2020). The implementation of the proposed method can be found in https://github.com/k-melnyk/graphKKE.git.

2.3.3 Evaluation

To evaluate the ability of the method to learn a low-dimensional representation of time-evolving graphs while maintaining metastability, we will use the following evaluation process. We first apply graphKKE to a time-evolving graph. We treat approximated m dominant eigenfunctions of the Koopman operator or Perron-Frobenious operator as a low-dimensional representation of the time-evolving graph. As was said above, a dimension of the new low-dimensional representation is defined by the number of dominant eigenvalues close to 1. Finally, we use the k-means clustering to identify clusters of points of the low-dimensional representation. In an ideal scenario, the resulting clusters should align with the metastable states observed in the initial time-evolving graph. As an evaluation metric, we will use the Adjusted Rand Index (ARI) which will be discussed in more detail further in this section. The utilization of the ARI allows us to evaluate and compare different approaches, parameter settings, or variations of the method in terms of their ability to capture and represent the metastable dynamics of time-evolving graphs accurately.

It is worth mentioning that our method works in an unsupervised manner and even though in our experiment we have ground-truth metastable states, in practice finding the number of metastable states and their location is what we aim at

obtaining from a low-dimensional representation.

k-means clustering

k-means clustering is one of the simplest and most popular unsupervised machine learning algorithms. The k-means clustering is based on the idea of identifying k numbers of centroids and then allocating every data point to the nearest cluster while keeping the centroids as small as possible.

Formally, the k-means clustering algorithm starts with choosing the number of clusters k. Then we initialize k cluster centroids randomly and each centroid represents the centre point of a cluster. Based on their distance we assign each data point to the nearest centroid. By computing the mean of all data points assigned to each cluster, we recalculate the centroid. We repeat the two last steps until convergence or until a stopping criterion is met. Note that k-means clustering is sensitive to the initial centroid positions. To mitigate this issue, we will run the algorithm several times with different random initializations. As a final result, we will average all runs of k-means.

Evaluating the performance of the clustering algorithm is not the same as in supervised learning where you have labelled data and can calculate accuracy, precision or recall. In the evaluation of the clustering algorithm, the objective is to obtain high intra-cluster similarity and low inter-cluster similarity. There are several clustering evaluation metrics available such as the Dunn index or Silhouette score which do not require any ground-truth labels for clusters. However, F-Measure, Rand Index or Adjusted Rand Index are evaluation metrics that require ground-truth labels. In our case, we have ground-truth labels for both synthetic data and real-world data, therefore we will use the Adjusted Rand Index which will be explained in more detail in the next paragraph.

Adjusted Rand Index

The Adjusted Rand Index (ARI) compares the similarity of two data clusterings or divisions. It considers both the agreement and the chance agreement between the clusters, yielding a score between -1 and 1, where 1 represents a perfect match, 0 indicates random clustering and negative values imply a disagreement worse than chance. The ARI compares the pairs of samples in the two clusterings. It measures the agreement in terms of three types of pairs: pairs that are in the same cluster in

both clusterings (true positives), pairs that are in different clusters in both clusterings (true negatives), and pairs that are in the same cluster in one clustering but in different clusters in the other clustering (false positives and false negatives). Before discussing the Adjusted Rand Index in more detail, it is necessary to first explain the Rand Index.

The Rand Index (RI) is defined as follows:

$$RI = \frac{TP + TN}{TP + FP + TN + FN}, \tag{2.29}$$

where TP is the number of true positives or the number of data point pairs that are clustered together in the predicted portion and in the ground-truth partition, TN is the number of true negatives, FP is the number of false positives or the number of data point pairs that are clustered together in the predicted partition but not in the ground truth partition and FN is the number of pairs of false negatives.

The Adjusted Rand Index is a correction of the Rand Index that measures the similarity between two clusterings by considering all pairs of samples and counting pairs that are assigned in the same or different clusters in the predicted and true clusterings. ARI is defined as:

$$ARI = \frac{RI - Expected_{RI}}{max(RI) - Expected_{RI}} \tag{2.30}$$

After we have defined our evaluation procedure which we will use in the experiments through the thesis, in the next section we will discuss synthetic and real-world data.

Comparative analysis

As a part of our evaluation procedure, we will also conduct a comparative analysis of the proposed method with other dimensionality reduction methods. The proposed approach is compared with one deep learning-based method, graph2vec, and the Weisfeiler-Lehman graph kernel. Graph2vec is an approach proposed to learn a low-dimensional representation of the entire static graph and it has been explained in Section 1.3. The Weisfeiler-Lehman graph kernel has been discussed in Section 2.2. Moreover, we will consider two variations of our approach: one with the WL graph kernel and one with the Gaussian kernel.

2.4 Synthetic data and Results

Most of the benchmark datasets such as those from chemo- and bio-informatics domains (Morris et al., 2020), can be represented only by static graphs. Since they do not contain time information and metastable behaviour, these datasets are not appropriate for our experiments. Therefore, we propose synthetic data that will be used to evaluate our method. The aim of this section is to explain how we construct the synthetic data with a comprehensible structure and known metastable states to estimate the performance of the proposed method. At the end of the section, we also present and discuss the results obtained with our method based on the evolution procedure mentioned above.

Energy potential

An energy potential, also known as a potential energy function or simply a potential, is a mathematical representation of a physical system's stored energy. It specifies how the system's energy varies as a result of its configuration or condition. A system's potential energy is critical in defining its equilibrium states, mobility, stability, and general behaviour. Physicists may examine the dynamics, forces, and interactions inside a system and anticipate its behaviour by evaluating potential energy. This concept will be utilized to generate synthetic datasets with metastable behaviour. When generating synthetic data, we will consider two different potential functions: double-well potential and lemon potential functions. The potential functions will be used as a base to incorporate metastable behaviour with different numbers of metastable states into a time-evolving graph. Let us consider a molecular-dynamics-inspired problem given by the stochastic differential equation

$$dX_t = -\nabla V(X_t)dt + \sqrt{2\beta^{-1}}dW_t \tag{2.31}$$

describing diffusion in the potential energy landscape given by $V(x)$. Here, β is the inverse temperature that controls the transition between states. The higher β, the less likely the transition from one state to another is. W_t is a standard Wiener process (Brownian motion). As was said above, we will utilize two following potential functions.

Double well potential

First, we will consider a time-evolving graph with two metastable states. In the context of the microbiome, it is frequently observed to have two metastable states, which can be understood as periods of illness and subsequent periods of recovery. To simulate a time-evolving graph with two metastable states, we consider the diffusion presented by Eq. 2.31 with a potential function

$$V(x) = \frac{x^4}{4} - \frac{x^2}{2}. \tag{2.32}$$

Figure 2.4 depicts an example of a trajectory of a particle in the double-well potential from Eq. 2.32. We will describe how we utilize this trajectory to construct a time-evolving graph with two metastable states later in the section, but first, we will present the second potential function, the so-called lemon potential that is used to incorporate more than two metastable states.

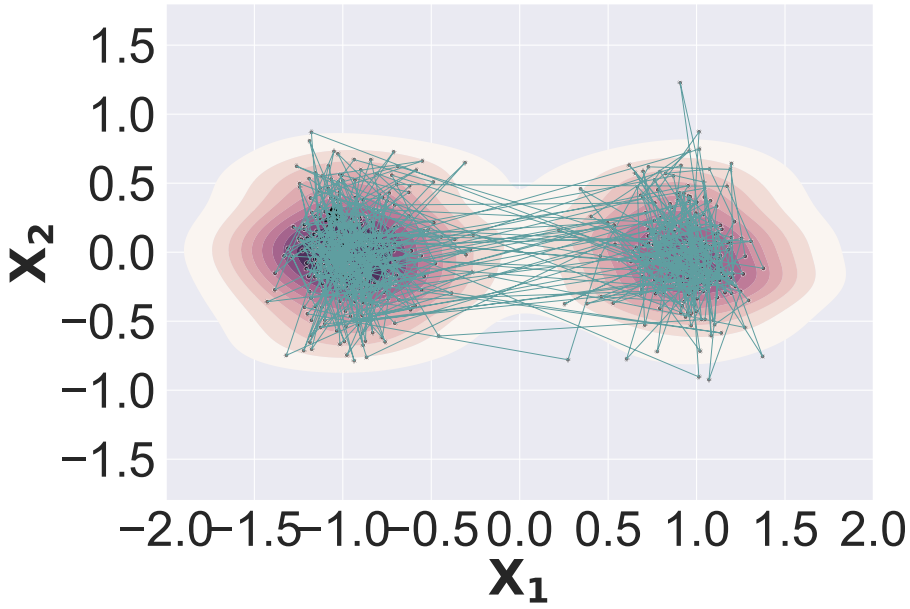


Figure 2.4: An example of a trajectory of a particle in the double-well potential. Points indicate the positions of the particle at time t and the blue lines show the movement of the particle from time point t to $t + 1$.

Lemon potential

In order to simulate time-evolving graphs with more than two metastable states, we will consider the diffusion defined by Eq.2.31 in the so-called lemon potential

$$V(x) = \cos(s \arctan(x_1, x_2)) + 10\left(\sqrt{x_1^2 + x_2^2} - 1\right)^2. \tag{2.33}$$

See (Klus et al., 2019b) for more details. Here, s denotes the number of wells or metastable states. The particle stays in one of the s wells for a relatively long time and then jumps to one of the neighbouring wells. An example of a particle trajectory in the lemon potential with five metastable states can be found in Figure 2.5.

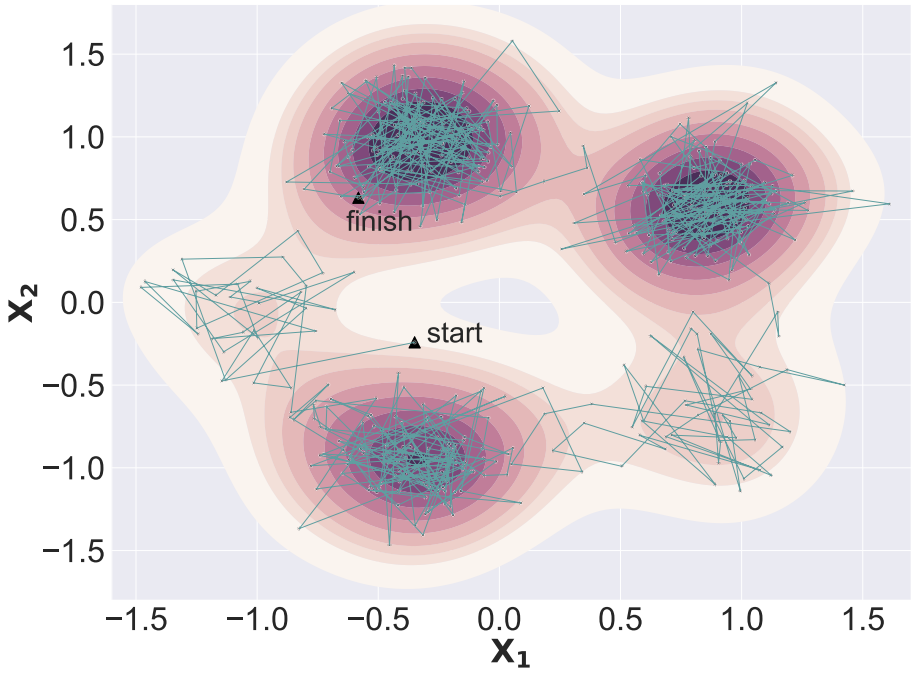


Figure 2.5: An example of a trajectory of a particle in the lemon potential with five metastable states. Points indicate the positions of the particle at time t and blue lines show the movement of the particle from time point t to $t + 1$.

By leveraging the concepts of the diffusion process in the double-well potential or in the lemon potential, we can effectively introduce metastable behaviour into a time-evolving graph. Further, it will explain how we simulate synthetic data.

2.4.1 Simulating synthetic data

As previously stated, there is a limited availability of datasets for evaluating our method. Therefore, we propose to generate synthetic data with metastable behaviour and ground-truth labels. In this section, we will outline the process of simulating synthetic data.

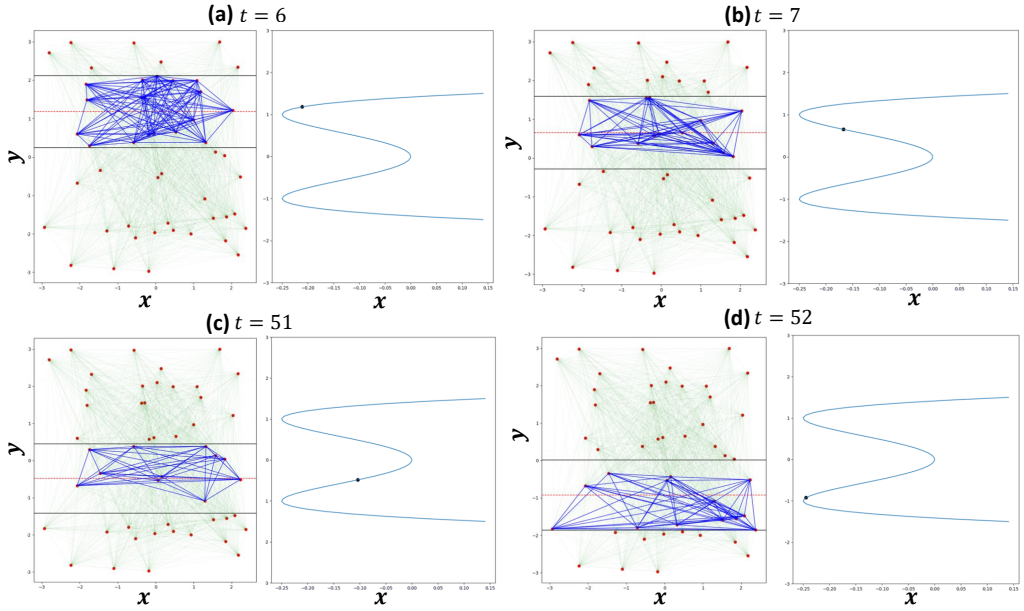


Figure 2.6: An illustration of the synthetic dataset with two metastable states at times **(a)** $t = 6$, **(b)** $t = 7$, **(c)** $t = 51$, and **(d)** $t = 52$. For each figure, the right subfigure depicts a particle in the double-well potential defined in Eq. 2.32, while the left subfigure shows the corresponding time-snapshot graph. The blue edges are randomly chosen and removed from the corresponding time-snapshot graphs. The node clusters in the rectangle are considered to be a structural graph pattern associated with a corresponding metastable state.

- First, we generate a trajectory $\mathcal{S} = \{(x_1^{(i)}, x_2^{(i)})\}_{i=1}^{T}$ using SDE (Eq. 2.31), and the corresponding potential function, the double-well potential (Eq. 2.32) or the lemon potential (Eq. 2.33).

- Then, we choose the number of nodes n, and assign random coordinates $(a_j, b_j), j = 1, \ldots, n$ to each of these nodes. The reason behind this is to determine the location of the discriminating characteristics associated with metastable states during the evaluation process.

- In the final step, we define discriminating structural features of the time-evolving graph for each state. Let G_0 be a complete time-snapshot graph at time $t = 0$. In the case of the s-well potential, we generate G_t, $\forall t, t = 1, \ldots, T$ by drawing a circle with the center at $(x_1^t, x_2^t) \in \mathcal{S}$ and the radius r and randomly removing edges between nodes that are inside the current circle. In addition, to address the noise in the real-world data, we also remove edges outside the current circle. For double-well potential, we remove edges between nodes, which satisfy $b_j > \frac{1}{T} \sum_{i=1}^{T} x_1^i$.

Figure 2.6 and Figure 2.7 illustrate examples of time-evolving graphs generated based on the above-mentioned process with the double-well potential (Eq. 2.32) and the lemon potential (Eq. 2.33), respectively. Figure 2.6 presents four time-snapshots graphs at different time points with a particle moving in the double-well potential. The edges in the blue colour are removed from the corresponding time-snapshot graph and are considered to be structural patterns of the time-evolving graph at the corresponding time point that are associated with each metastable state. Additionally, Figure 2.7 depicts two time-snapshots graphs at different time points. The edges within the circular regions are removed from the corresponding time-snapshots graphs, and the clusters of nodes within the circles are regarded as structural patterns of the time-evolving graph linked to metastable states. In the right images, it can be seen positions of a particle moving a lemon potential with five metastable states.

2.4.2 Experiments and Results

After we have introduced our method in Section 2.3, defined the evaluation process in Section 2.3.3 and explained the idea behind synthetic data in Section 2.4, we will finally introduce our results. In this section, we aim to evaluate the proposed method on synthetic time-evolving graphs with metastability. Our main objective is the following. Given a time-evolving graph, the ultimate goal is to find a number of metastable states and their locations in time-evolving graphs. Besides, we conduct a comparison of our method with the Weisfeiler-Lehman kernel and deep learning-based method graph2vec. Moreover, we investigate what impact various graph kernels such as WL or Gaussian kernels have on the performance of the proposed method.

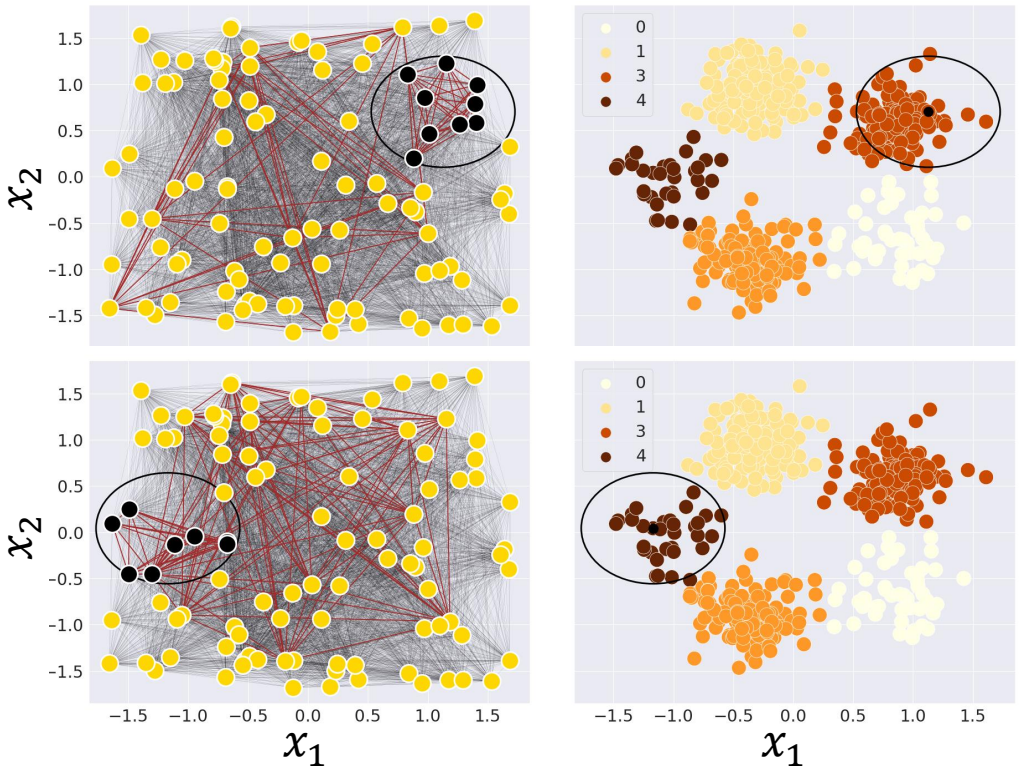


Figure 2.7: An illustration of the presented synthetic data with five metastable states at times **(a)** $t = 0$, **(b)** $t = 256$. For both **(a)** and **(b)**, the right images are positions of a particle in the 5-well potential, which are clustered into 5 sets with k-means and the left images show time-snapshot graphs G_0 and G_{256}. The edges in the red colour are removed from the graph. The nodes and removed edges in the circles are considered graph patterns associated with corresponding metastable states.

Experimental setup

In order to test the performance of the method proposed in Section 2.3 and compare the result to other baselines models, we generate the synthetic data described in Section 2.4 with different configurations of interest such as the number of vertices n, the number of time steps T, and the number of states s. The datasets are summarized in Table 2.2. For each dataset, we set the out-state probability to 0.1. The out-state probability is the probability of removing edges outside the corresponding circle. For the analysis of metastable behaviour, we apply graphKKE with the Weisfeiler–Lehman graph kernel with a number of iterations $h = 1$ and regularization parameter $\eta = 0.1$. In order to have ground truth labels/states of $\mathcal{G}$, we apply k-means clustering to the realization $\mathcal{S}$ of SDE defined in Eq. 2.31. For the Weisfeiler–Lehman kernel,

Table 2.1: Hyperparameters for graphKKE used in the comparative analysis, where σ is the bandwidth of Gaussian kernel, h the number of iterations in the WL kernel, and η the regularization parameter of Tikhonov regularization.

Dataset	σ	h	η
pos_5DynG-100	10	1	0.1
pos_5DynG-200	100	1	0.5
pos_3DynG-300	100	1	0.1
MovingPic	100	1	0.5

Table 2.2: Statistics of each dataset used in the Chapter 2.

Name	**#Vertices**	**#Edges (avg.) ± std.**	**#Time steps**	**#States**
pos_5DynG-100	100	4851±43.68	500	5
pos_5DynG-200	200	19516±119.54	1000	5
pos_3DynG-300	300	44051±219.05	500	3
MovingPic	919	10602±7266.39	658	2
CholeraInf	96	106±41.28	34	2

the initial set of vertex labels l_0 is defined to be $\{0, 1, 2, \ldots, n\}$.

In the comparative analysis, as was already mentioned, we aim to compare graphKKE to several state-of-the-art representation learning and graph clustering approaches. The proposed approach with two different graph kernels — Gaussian and Weisfeiler–Lehman kernels — is compared with graph2vec (Narayanan et al., 2017) and the original WL kernel (Shervashidze et al., 2011). The main idea of graph2vec is explained in Section 1.3 and the WL kernel is discussed in Section 2.2. Since the analysis is done for the graph clustering task, we apply k-means to the resulting embedding vectors of every approach. The embedding dimensions of $\{5, 64, 128, 1024\}$ were chosen for graph2vec. The final embedding is chosen based on the best model run. The hyperparameters of graphKKE were chosen empirically[1] and can be seen in Table 2.1. The choice of σ for the Gaussian kernel is critical for the performance of graphKKE. The theoretical justification of applying the Gaussian kernel to the graphs-structured data and the optimal choice of σ is beyond the scope of this thesis.

[1]The combinations of hyperparameters with the biggest spectral gap were used.

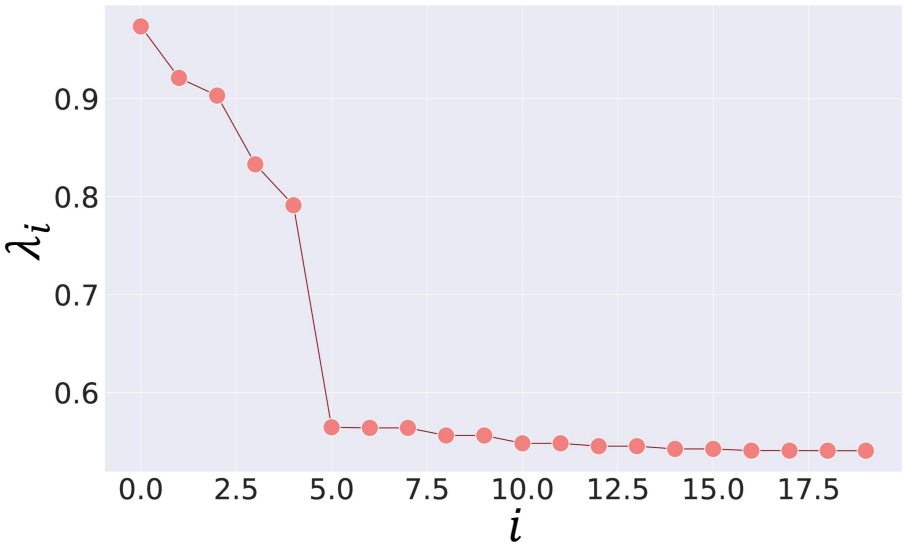


Figure 2.8: The eigenvalues of the Koopman operator approximated by graphKKE for pos_5DynG-100 dataset. The large spectral gap after the fifth eigenvalue reveals the presence of the 5 metastable states in the dataset.

Analysis of metastability

We visualize approximated eigenvalues and eigenfunctions only for the pos_5DynG-100 dataset. The eigenvalues of the Koopman operator approximated with graphKKE are shown in Figure 2.8. A spectral gap after the fifth eigenvalue indicates that the time-evolving graph $\mathcal{G}$ contains $s = 5$ metastable states and $\mathcal{G} = \mathcal{G}_0 \cup \cdots \cup \mathcal{G}_4$. Since all information about the long-term behaviour of the time-evolving graph is contained within the eigenfunctions of the Koopman operator associated with s dominant eigenvalues close to 1 (in our case $s = 5$), the final dimension m of a low-dimensional vector is defined by the number of these eigenfunctions. Thus, for the pos_5DynG-100 dataset, each time-snapshot graph of $\mathcal{G}$ is embedded into a new vector space $\mathbb{R}^m$ with $s = m = 5$. An illustration of the embedded time-series data $\varphi(\mathcal{G})$, where $\varphi(\mathcal{G}) = \sum_{i=0}^{4} c_i \varphi_i(\mathcal{G})$ with $c_0 = 0, c_1 = \frac{1}{2}, c_2 = \frac{1}{8}, c_3 = \frac{1}{8}, c_4 = \frac{1}{4}$, is shown in Figure 2.9.The coefficients are chosen to make the visualization of the five metastable states clear and distinguishable.

After we obtain the low-dimensional representation of the time-evolving graph, it can serve as an input for various machine learning approaches or to further analyze the dynamics of the time-evolving graph by constructing a Markov State Model in which the main metastable states form the states of a Markov chain. By employing this lower-dimensional representation, we can explore a broader spectrum of machine

learning methodologies that can leverage the insights gained from the reduced space. This can include tasks such as clustering, classification, or even anomaly detection. Additionally, the reduced representation maintains the essential characteristics of the initial dynamics of the time-evolving graph while potentially reducing the complexity of computations.

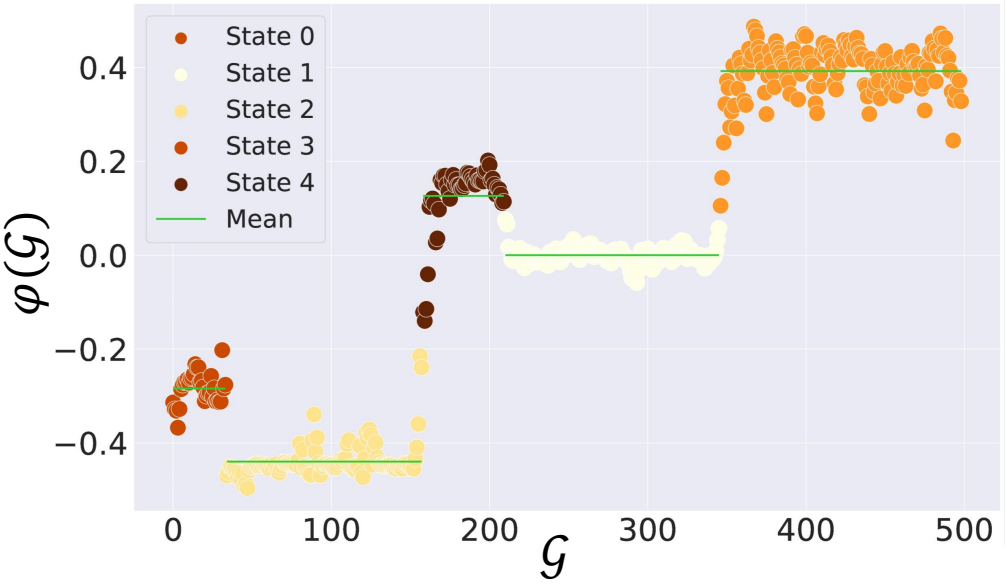


Figure 2.9: An illustration of the embedded time-evolving graph, where $\varphi(\mathcal{G}) = \sum_{i=0}^{4} c_i \varphi_i(\mathcal{G})$ and $\varphi_i(\mathcal{G})$ are dominant eigenfunctions of the Koopman operator approximated by graphKKE. There are 5 distinct level sets corresponding to the 5 metastable states.

Table 2.3: Adjusted Rand Index (ARI) for the comparative analysis on the graph clustering task for *the synthetic datasets*. Higher ARI corresponds to greater accuracy in correctly identifying the ground truth states. It can be seen that the combination of graphKKE with the Weisfeiler-Lehman kernel outperforms other methods.

Dataset	graph2vec	WL kernel	(graphKKE, WL)	(graphKKE, Gk)
pos_5DynG-100	0.49	0.36	0.99	0.96
pos_5DynG-200	0.20	0.49	0.92	0.87
pos_3DynG-300	0.22	0.40	0.96	0.94

Comparative analysis

The key aim of the comparative analysis is to show how the low-dimensional representation can be used for the further analysis of microbiome dynamics. Here, we try to find the location of metastable states using the low-dimensional representation or eigenfunctions. This problem can be formulated as a clustering.

The result of the comparative analysis can be found in Table 2.3. For graph2vec the embedding dimension of five was used as a dimension with the best ARI to compare its result with the results of other approaches. We observe that both graph2vec and WL kernel perform poorly on the synthetic datasets. One reason for the poor embedding is that these two methods do not take into account the time information which is crucial in time-evolving graphs with metastability.

Additionally, the high ARI scores of graphKKE indicate that it can learn a low-dimensional representation in such a way that the metastable behaviour is maintained in a lower-dimensional space. The result of clustering on a low-dimensional representation obtained with graphKKE with the Gaussian kernel indicates that the Gaussian kernel can be used to graph-structure data. However, in cases of more complex changes in the graph structure, the Gaussian kernel might not be able to capture them.

2.5 Microbiome data and Results

The microbial communities living in the human intestine can have a profound impact on our well-being and health. However, we have a limited understanding of the mechanisms that control this complex ecosystem. Moreover, microbial communities are not static over time as we discussed in Introduction 1.1.2. Therefore, the analysis of the microbiome dynamics is essential for a better understanding of processes occurring in the microbiome constitution. In the previous section, we demonstrated the capability of our method to identify metastable states in synthetic data. Through a comparative analysis, we also established that graphKKE surpasses specific other methods in graph clustering tasks, which are relevant to locating metastable states in the time-space domain. In this Section, our aim is to show that it is feasible to apply our method to understand these mechanisms that control complex microbiome ecosystems. One of our main objectives in this thesis is to make the analysis of the microbiome easier by embedding a time-evolving graph constructed from the micro-

biome into a low-dimensional representation. This low-dimensional representation can further be used to analyze the microbiome dynamics.

We begin this Section by introducing microbiome datasets and their preprocessing. Specifically, we will employ the MovingPic dataset, which comprises samples from various body sites, as well as the CholeraInf dataset, consisting of periods of cholera infection and subsequent recovery in multiple individuals. The subsequent section will provide a more comprehensive overview of each dataset. Finally, we will present our results following the same structure as it was discussed in Section 2.3.3.

2.5.1 Microbiome datasets

MovingPic dataset

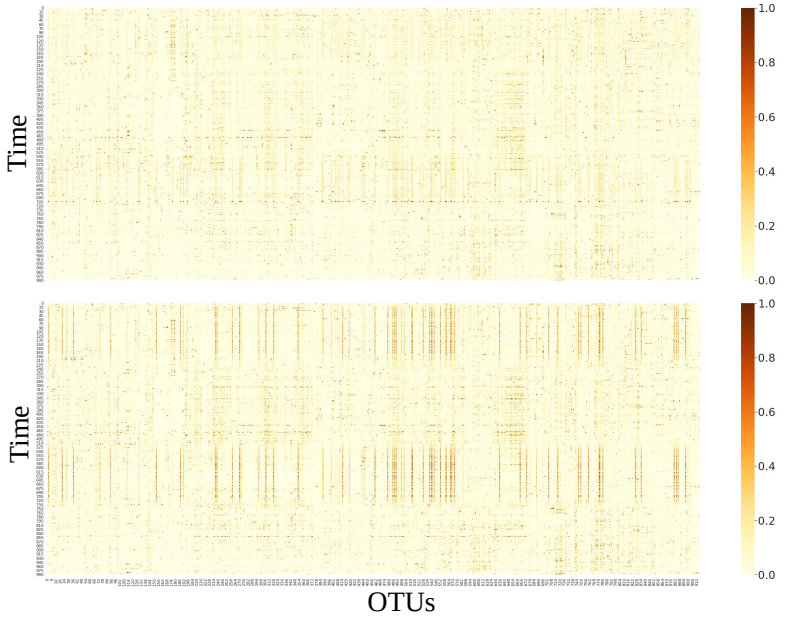


Figure 2.10: A visualization of the real-world MovingPic dataset, where columns are the concentrations of particular species over time. The darker colour specifies the higher concentration. **Top:** shows the original MovingPic dataset and **Bottom:** the dataset with added 5% noisy signal.

The MovingPic dataset was first presented in (Caporaso et al., 2011). In this study, two healthy subjects, one male and one female, were sampled daily at three body sites (gut, skin, and mouth) for 15 months and for 6 months, respectively. This study aimed at identifying differences in microbiota between body sites and individuals. Their results showed that despite relatively stable differences between body sites

and individuals, there was, however, variability in an individual's microbiota across months, weeks, and days. It is interesting that according to their results, only a small amount of the total taxa found within a single body site appear to be present across all time points. For our experiments, we use the microbiome profile only from the skin. Moreover, since the MovingPic dataset does not have any perturbations such as antibiotics exposure or diseases, we add an artificial noisy signal to incorporate metastable behaviour. A practical justification for adding noise to the signal is that the human microbiome might react not only to major perturbations such as diseases or antibiotics exposure but also to some short-term daily fluctuations such as changes in lifestyle or stress. Moreover, the noise will be added to test the robustness of graphKKE.

We will start by explaining the operational taxonomical unit (OTU) table. The OTU table is a tabular representation of microbial diversity data obtained through high-throughput sequencing techniques. It provides information about the abundance or presence of different taxa in a given sample. Formally, we define it as a $D \in \mathbb{N}^{T \times p}$ matrix with T being the number of time points and p being the number of OTUs. Next, we will explain how we preprocess the OTU table, integrate a signal and finally, construct a time-evolving graph. Let $d_i = [d_i^0, d_i^2, \ldots, d_i^{T-1}]$ be the T-dimensional column vector of OTU counts of the ith species. OTUs with less than 30% of total reads are removed from the matrix D. Then, we randomly choose 100 OTUs that are used to add the noisy signal. The vector of length T is constructed using a sine wave function:

$$z = R \sin\left(\frac{2\pi t}{\omega}\right)$$

and then for each $i, i = 0, \ldots, 100$, we compute new OTU counts d_i

$$d_i = d_i + \max(0, z + \epsilon w z),$$

where $w \sim \mathcal{N}(0, 1)$ and ϵ is the level of Gaussian noise. We set ϵ to one of $\{0, 0.05, 0.3\}$ and will report results for each ϵ. Also, let $d^t = [d_1^t, d_2^t, ..., d_p^t]$ be the p-dimensional row vector of OTU counts at time point $t, t = 0, ..., T-1$. The raw OTU counts are typically normalized by the total cumulative count $c^t = \sum_{i=1}^{p} d_i^t$ in order to account for the different sequencing depths (Lo and Marculescu, 2019). Thus, the normalization of d^t by the total cumulative count results in the relative abundance

vector:

$$x_t = \left[\frac{d_1^t}{c^t}, \frac{d_2^t}{c^t}, \ldots, \frac{d_p^t}{c^t}\right]$$

for each time point $t, t = 0, \ldots, T-1$.

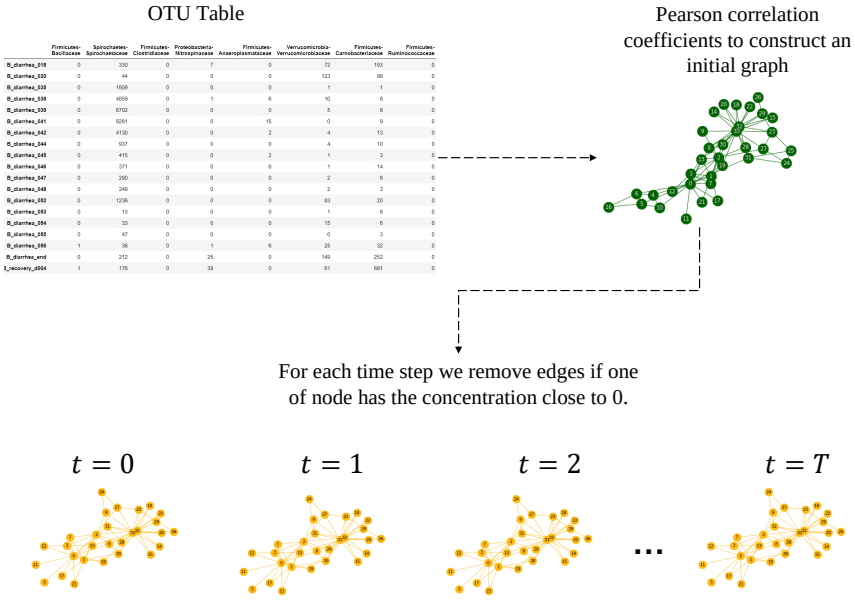


Figure 2.11: The construction of a time-evolving graph from the OTU table. 1) Pearson correlation coefficients are computed in order to construct an initial co-occurrence graph at $t = 0$. 2) $\forall t$, we remove edges from the initial time-snapshot graph if the concentration of a species is close to zero.

Given the OTU table D, the next step is the construction of a time-evolving graph $\mathcal{G} = (G_0, ..., G_{T-1})$. The overview of the construction of a time-evolving graph from the microbiome data can be found in Figure 2.11. First of all, we compute the Pearson correlation coefficients for each pair of OTUs (d^i, d^j), with $i, j = 1, ..., p$ in order to define an initial co-occurrence graph. It is worth noting that in Section 1.1, we have highlighted different challenges that the microbiome analysis usually encounters. One of these challenges is compositionality which is not captured by Pearson correlation coefficients. However, since the key focus of this thesis is not how to construct a time-evolving graph from the microbiome, but rather to propose methods for the analysis of the microbiome data, we choose not to explore various approaches for constructing networks from the microbiome. We select a threshold of 0.5 such that edges with the Pearson coefficient greater than 0.5 or less than -0.5 are considered to be strongly correlated and remain in G_0. Edges with the Pearson

correlation coefficient in the range $[-0.5; 0.5]$ are removed from the initial graph. Furthermore, in order to construct time-snapshot graphs for each $t = 0, \ldots, T-1$, we use the OTU table D. If the OTU count for the current vertex is zero, we remove edges connecting this vertex and its neighbouring vertices. The statistics of the pre-processed data can be seen in Table 2.2. Moreover, we define $\widehat{\mathcal{G}}_t = \mathcal{G}_{t+\tau}$. That is, for the chosen lag time τ (in our case $\tau = 1$), $\mathcal{G} = (G_0, \ldots, G_{T-2})$ and $\widehat{\mathcal{G}} = (G_1, \ldots, G_{T-1})$. From the two time-evolving graphs $\mathcal{G}$ and $\widehat{\mathcal{G}}$, we compute the Gram matrices $K_{\mathcal{G}\mathcal{G}}$ and $K_{\mathcal{G}\widehat{\mathcal{G}}}$ using the Weisfeiler–Lehman kernel.

Cholera infection

Another real-world microbiome data that we analyze in this section originates from a study about recovery from Vibrio cholerae infection (Hsiao et al., 2014). Fecal microbiota was collected during acute diarrhea and recovery periods of cholera in a cohort of seven Bangladeshi adults. In our experiments, we chose one patient, since there is variation in the constituents of the gut microbiota among individuals (Durack and Lynch, 2019) and thus, it can bias the result of detecting the metastable states such as diarrhea and recovery periods. Moreover, we will conduct our experiments on a smaller cholera infection dataset. The original dataset contains around 343 different species after filtering and rarefaction and will be left for future work. The pre-processed OTU table was obtained from Zackular et al. (2015). The aim is to determine if there are metastable states in this data and if possible, the number of metastable states and their locations.

The time-evolving graph from the given OTU table is constructed in the same way as for the MovingPic dataset using the relative abundance vector and Pearson correlation coefficients. In the real-world microbiome dataset, perturbations do not always shift OTU counts to zero. Therefore, the question of how to properly construct time-evolving graphs such that both metastable behaviour and associations between microbes are taken into consideration needs to be considered in future work.

2.5.2 Experiments and Results

Experimental setup.

We define $\widehat{\mathcal{G}}_t = \mathcal{G}_{t+\tau}$. That is, for the chosen lag time $\tau = 1$, $\mathcal{G} = (G_0, \ldots, G_{T-2})$ and $\widehat{\mathcal{G}} = (G_1, \ldots, G_{T-1})$. From the two time-evolving graphs $\mathcal{G}$ and $\widehat{\mathcal{G}}$, we compute

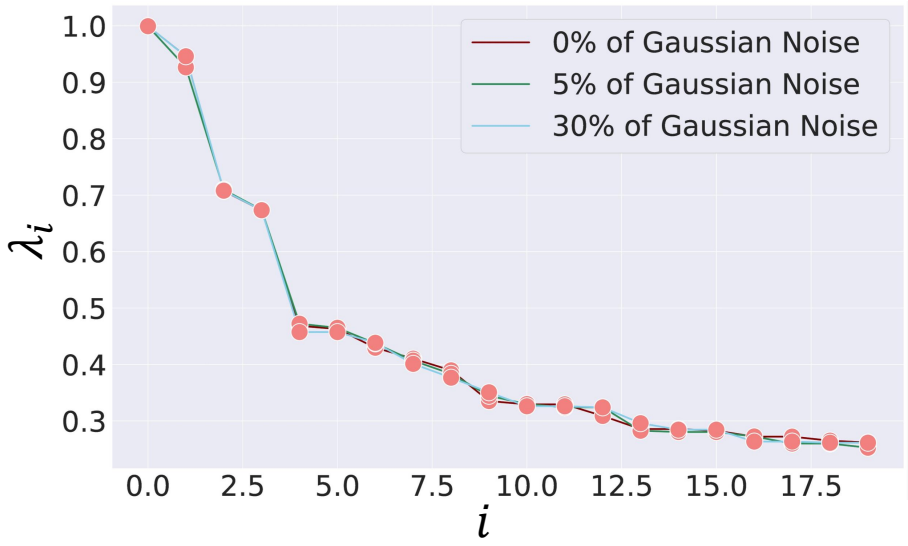


Figure 2.12: The eigenvalues of the Koopman operator approximated by graphKKE for different percentages of Gaussian noise added to the MovingPic dataset. The spectral gap after the second eigenvalue indicates the presence of 2 metastable states.

the Gram matrices $K_{\mathcal{G}\mathcal{G}}$ and $K_{\mathcal{G}\widehat{\mathcal{G}}}$ using the Weisfeiler–Lehman kernel. In the case of the MovingPic dataset, we set the number of iterations to be $h = 1$ and the regularization parameter to be $\eta = 0.9$. In the case of the Cholera infection dataset, the number of iterations is 5 and the regularization parameter is 0.1. The statistics of each real-world dataset can be found in Table 2.2.

Analysis of metastability

The eigenvalues detected by graphKKE for the MovingPic dataset with different percentages of Gaussian noise are shown in Figure 2.12. The gap after the second eigenvalue and the values of these eigenvalues close to 1 implies the presence of two states in the time-evolving graph $\mathcal{G}$. The spectral gap after the fourth eigenvalue indicates the presence of four states but we are not aware of the biological interpretations of the second two states since the original study does not mention any potential perturbations. Future work can be focused on the analysis of these other two metastable states. The experiment also shows that graphKKE is robust to the noise in the data since regardless of the noise level in the data the method detects two metastable states. In order to find the location of the states, we cluster time-snapshot graphs into two states using k-means applied to the two normalized eigenfunctions associated with two dominant eigenvalues with the number of clusters set to 2. Fig-

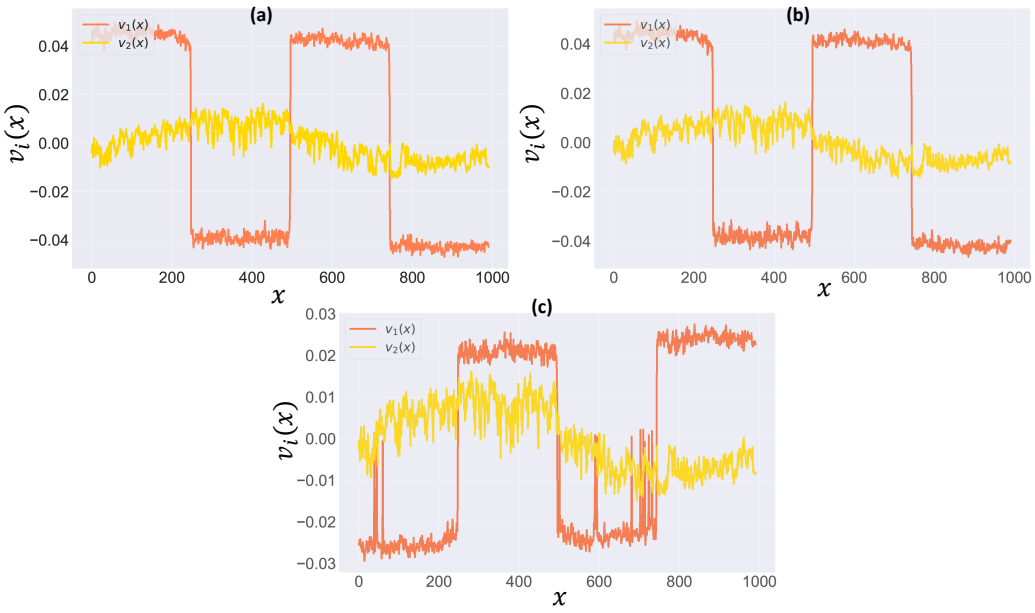


Figure 2.13: The approximated eigenvectors of the Koopman operator associated with two dominant eigenvalues for different levels of noise added to the MovingPic dataset: **(a)** without noise **(b)** with 0.05 noise and **(c)** with 0.3 noise.

ure 2.13 depicts two eigenvectors associated with two dominant eigenvalues close to 1 for each level of Gaussian noise. We can see that graphKKE is relatively robust to the noise in the data. Moreover, the first eigenvector has a constant value close to 0 and the second eigenvector indicates the presence of two states: one in 0.04 and another one in -0.04.

For the CholeraInf dataset, the resulting eigenvalues are shown in Figure 2.14. Two dominant eigenvalues close to 1 imply that the time-evolving graph $\mathcal{G}$ contains two metastable states. In the comparative analysis, we will show that these two metastable states correspond to the ground truth infection/recovery periods of the dataset. Moreover, the eigenfunctions associated with these two dominant eigenvalues (Figure 2.14 (Right)) contain all information about the long-term behaviour of the time-evolving graph $\mathcal{G}$ and using them as a low-dimensional representation we can further analyze the cholera dataset with the aid of time-series methods which work with vector-structured data. For example, one can cluster the data into two clusters, predict the state at the next time point or we can find the probability of $\mathcal{G}$ returning to the diarrhea state if a person continues living in this area, however, this is out of scope of this thesis.

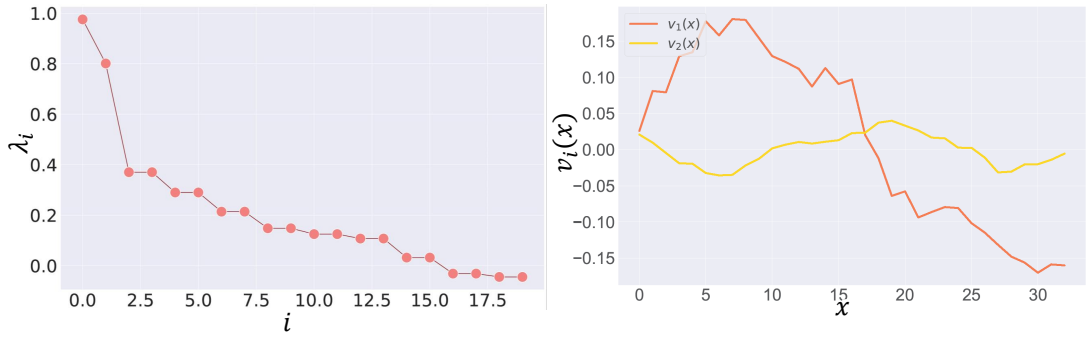


Figure 2.14: **Left:** The large spectral gap after the second eigenvalue of the Koopman operator approximated by graphKKE indicates that the CholeraInf dataset consists of 2 metastable states. **Right:** The approximated eigenvectors of the Koopman operator associated with two dominant eigenvalues.

Comparative analysis

For the comparative analysis, we will utilize the dominant eigenfunctions of transfer operators for both datasets. Applying k-means to eigenfunctions associated with two dominant eigenvalues, we can find the location of metastable states in $\mathcal{G}$. Moreover, in order to estimate whether the resulting low-dimensional representation maintains the dynamics of the time-evolving graph, we will compare the metastable states, which we obtained by clustering eigenfunctions, with the initial time periods of diarrhea and recovery. The ARI is shown in Table 2.4 for both datasets. Similar to the results on synthetic datasets, graph2vec, and WL kernel perform poorly compared to both graphKKE with WL and graphKKE with Gaussian kernel. It is interesting that ARIs for graphKKE with WL kernel and graphKKE with Gaussian kernel are almost the same.

Table 2.4: Adjusted Rand Index (ARI) for the comparative analysis on the graph clustering task for *the real-world datasets*. Higher ARI corresponds to greater accuracy in correctly identifying the ground truth states. It can be seen that the combination of graphKKE with the Weisfeiler-Lehman kernel outperforms other methods.

Dataset	graph2vec	WL kernel	(graphKKE, WL)	(graphKKE, Gk)
MovingPic	0.42	0.56	1	0.99
CholeraInf	0.29	0.66	0.88	0.87

2.6 Discussion

The large variety of species and complex interactions in the microbiome makes it challenging for researchers to analyze the responses of the microbiome to different perturbations such as diseases or antibiotic exposures and its influence on human health. However, most studies aiming at understanding these dynamics are primarily focused on statistical constitution analysis ignoring more complex interactions that can be described in the form of a time-evolving graph. One solution is to represent each time-snapshot graph of the time-evolving graph as a fixed-length feature vector instead of trying to analyze the complex metastable dynamics of the microbiome composition in the high-dimensional space. However, many existing approaches learn the embedding either of the static graphs or of the substructures such as nodes, edges, or subgraphs, whereas for some systems it is of great importance to embed the entire time-snapshots of the time-evolving graph into a low-dimensional space preserving the global temporal mechanisms such as metastability.

Therefore, in this Chapter, we introduced an unsupervised approach (i.e., class labels of a single time-snapshot graph are not required to learn the embedding) for learning a mapping that embeds a time-snapshot graph of a time-evolving graph exhibiting metastable behaviour as points in a low-dimensional vector space. The proposed method utilizes two key concepts, namely, transfer operators and graph kernels. Transfer operators help to capture the dynamical properties of the initial high-dimensional complex system (time-evolving graph in this case) and graph kernels extract the structural properties associated with each time-snapshot graph. Both concepts help us to learn a low-dimensional representation of the time-evolving graph in such a way that the initial dynamics such as metastability are preserved in a new space. Furthermore, we introduced synthetic datasets that make use of the concept of potential functions. We were able to construct synthetic time-evolving graphs with different configurations such as the number of metastable states, and the number of nodes/edges. Finally, we presented real-world microbiome datasets for which we aim to extract new biological insights using the proposed method.

Our experiments on the synthetic and real-world data have shown that our approach is capable of learning a low-dimensional representation of the time-evolving graph that preserves the metastable behaviour. The points of the low-dimensional representation can then be clustered in order to identify the location of metastable

states (e.g. time points corresponding to cholera or recovery period). Moreover, one can also analyze the dynamics occurring in the time-evolving graph (e.g., the probability of jumping from one state to another or the probability that the graph will return to one of the states) and apply different machine learning techniques. Since we are dealing with graph-structured data, which usually represents the interactions between objects, we can extract structural information pertaining to particular states. The latter is beneficial in the case of biological interactions such as microbiome data, where it is crucial to understand the differences between states (e.g., healthy or ill). To this end, experimental results have shown that our approach can outperform several state-of-the-art methods for the representation learning of graphs. For instance, the comparative analysis has shown that applying only the Weisfeiler–Lehman kernel to the time-evolving graph is not sufficient to capture the underlying dynamical graph patterns and consequently, to detect the metastable sets.

We have shown that graph kernels are not only a powerful tool for analyzing static graphs but also for analyzing time-evolving graphs. The transfer operator approach in combination with graph kernels yields a method capable not only of extracting structural information in each time-snapshot graph of the time-evolving graph but also of identifying the evolution patterns, which may exist in time-evolving graphs with metastability over long periods of time.

In conclusion, it is important to acknowledge the limitations of our proposed method. One significant limitation is an inability to extract structural graph features that are linked to each metastable state. This is crucial for understanding changes occurring in the microbiome compositions under the influence of external perturbations. We will discuss this in more detail in a later chapter.

Chapter 3

Understanding microbiome dynamics via graph representation learning

Understanding and analyzing complex systems such as the microbiome with a large number of interactions and high dimensionality is a challenging task. In Chapter 2, we proposed a method that simplifies the analysis of such complex systems, in particular the system of microbial species, by learning a low-dimensional representation of a time-evolving graph while maintaining the metastable behaviour in a new space. However, the complexity of interactions, the large number of time steps, and high dimensionality require not only alternative approaches such as the embedding of the system of microbial into the low-dimensional space but also extracting microbial biomarkers such as edge or node clusters which are associated with changes in microbial composition as it was mentioned in Section 1.1. Unfortunately, the graphKKE method introduced in the previous chapter does not allow transitioning from a low-dimensional space back to the space of the time-evolving graph. This means that it is impossible to extract microbial biomarkers while learning a low-dimensional space. In Chapter 4 we will propose an approach that might be used for this purpose, however in case of more complex structural features of the time-evolving graph associated with changes in microbial compositions, this approach will not work.

In this chapter, we will continue studying how to embed the time-evolving graph into a low-dimensional space while maintaining the metastable behaviour. However, here we study the application of modern deep learning techniques to resolve the problem stated above. Our approach begins by utilizing the Transformer architecture to compute the embedding of a time-snapshot graph. The Transformer's attention mechanism allows us to capture the relationships and dependencies between different nodes or edges in the graph, enabling a comprehensive representation of the system's dynamics at a particular time point. Furthermore, to ensure that the representations of consecutive time-snapshot graphs, which exhibit similar metastable behavior, are close to each other, we employ contrastive learning. Contrastive learning allows us to emphasize the similarities and differences between time-snapshot graphs, enabling the extraction of crucial information related to metastable states. In addition to presenting our proposed approach, we also provide a comparative analysis with the kernel-based method presented in Chapter 3. By contrasting the performance and efficacy of our interpretable graph representation model with the kernel-based method, we aim to showcase the advantages and novel insights that can be gained through the application of modern deep learning techniques.

Our main contribution is presenting a model that learns a low-dimensional rep-

resentation of the time-evolving graph with metastable behaviour in an unsupervised manner. We show in experiments that the metastability governing the time-evolving graph is preserved by the model. By interpreting the output of the model with respect to the input, we demonstrate that it is feasible to extract topological features of the time-evolving graph, which define each metastable state. These features can be used to identify a set of microbes that drive the microbiome constitution to undergo transitions from one metastable state to another.

3.1 Transformer

The Transformer is a deep learning architecture that has had a large impact on various natural language processing (NLP) tasks. It was introduced in the paper "Attention Is All You Need" by (Vaswani et al., 2017), and it revolutionized the way NLP models process and understand sequential data. Moreover, the Transformer has been attracting much attention in other domains such as computer vision (Dosovitskiy et al., 2020) and graph representation learning (Velickovic et al., 2017; Dwivedi and Bresson, 2020).

Before transformers were invented, the traditional approaches for sequential data included recurrent neural networks (RNNs) and long short-term memory networks (LSTM) which are costly and time-consuming to train on large datasets. These approaches suffer from sequential processing bottlenecks, causing limitations in capturing long-range dependencies efficiently. In order to overcome these limitations, attention and self-attention mechanisms have been proposed. These components allow more efficient parallel processing, making it possible to scale the speed and capacity of sequential deep-learning models. In technical details, a transformer architecture consists of an encoder and a decoder, both of which are made up of multiple layers of self-attention and feed-forward neural networks. The encoder takes the input sequence and produces a fixed-length context vector, which is then used by the decoder to generate the output sequence. The key innovation of the transformer is the use of self-attention mechanisms, which allow the model to weigh the importance of different parts of the input sequence when making predictions. For instance, in the context of natural language processing, this allows the model to learn relationships between words in a sentence, regardless of their position in the sentence.

3.1.1 Transformer Architecture

In this section, we discuss each of the key components of the Transformer, the motivation behind the creation of the Transformer, and why the Transformer has had such success. The Transformer is introduced as a novel encoder-decoder architecture built with multiple blocks of a self-attention mechanism. Figure 3.1(a) depicts the original architecture of the Transformer.

As most competitive neural sequence transduction models have an encoder-decoder structure (Vaswani et al., 2017), the Transformer is not an exception. The encoder takes an input sequence $(x_1, \ldots, x_n)$ and maps it to a continues-values sequence $\mathbf{z} = (z_1, \ldots, z_n)$. Given $\mathbf{z}$, the decoder then generates an output sequence $(y_1, \ldots, y_n)$ of symbols one element at a time. This is a common structure of all neural machine translation models. However, the architecture of the Transformer has three components that make it successful. We will discuss them further in the thesis and begin with the encoder proceeding with each component of the Transformer step by step.

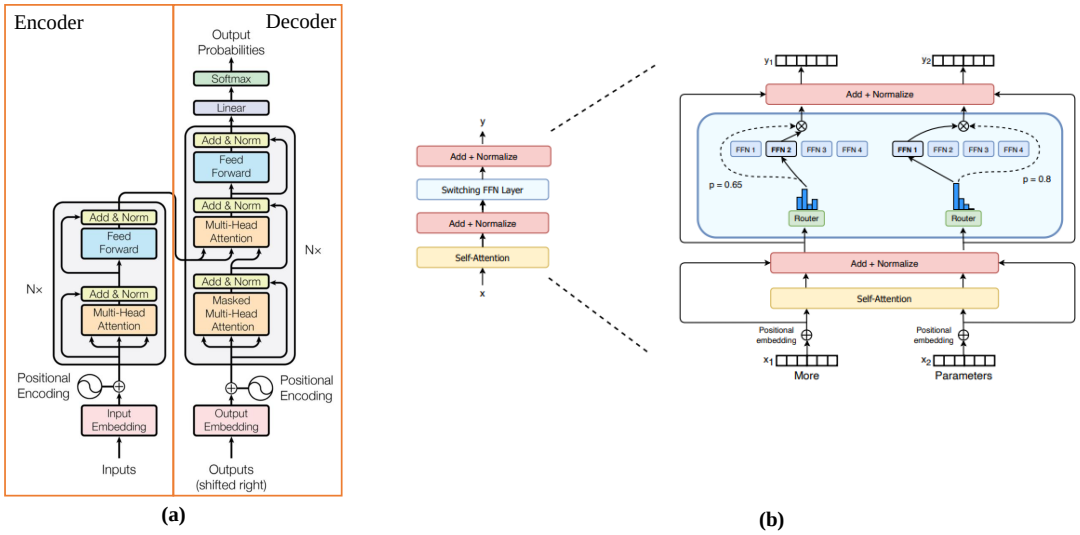


Figure 3.1: **(a)** The architecture of original Transformer. The image is taken from (Vaswani et al., 2017). **(b)** The Switch Transformer. Image from the original paper (Fedus et al., 2022).

Encoder

The encoder is responsible for transforming the input sequence into a latent representation. The encoder of the original Transformer is composed of a stack of $N = 6$ identical layers. Each layer has two sub-layers (the encoder is shown in Figure 3.1 on the left). One sub-layer is a multi-head self-attention mechanism, which will be discussed later, and the second sub-layer is a fully connected feed-forward network. Each of the two sub-layers is followed by a residual connection (He et al., 2016) and layer normalization (Ba et al., 2016).

Decoder

The decoder is designed to generate the target sequence, leveraging the information processed by the encoder. The decoder is also composed of a stack of $N = 6$ identical layers. In addition to the two sub-layers similar to the encoder, the decode has a third sub-layer which performs multi-head attention over the output of the encoder stack. This layer is also called "encoder-decoder attention" and it enables the decoder to access information from the input sequence through the attention mechanism. Residual connections and layer normalization are employed after each sub-layer as it is done in the encoder. To prevent the decoder from attending future positions during training, a masking technique is applied to the self-attention mechanism in the decoder part. The masking ensures that each position can only attend previous positions in the sequence.

Positional Encoding

The Transformer does not contain any recurrence layers that could capture the order of words in a sentence. Therefore, information about the relative or absolute position of words in the sequence must be injected to make use of the order of the sequence. The authors employ absolute positional encoding (PE) to the input embeddings before the encoder and decoder stacks. The positional encodings have the same dimension as the embedding. The original Transformer uses sine and cosine functions of different frequencies:

$$p_{pos,2i} = \sin\left(\frac{pos}{10000^{\frac{2i}{d_m}}}\right) \tag{3.1}$$
$$p_{pos,2i+1} = \cos\left(\frac{pos}{10000^{\frac{2i}{d_m}}}\right),$$

where pos, i and d_m denote the position of the node in the time-snapshot graph, the dimension in the positional encoding, and the dimension of node embedding, respectively.

Various strategies for positional encoding have been proposed. For example, one way of representing absolute positions is to learn a set of positional embeddings for each position (Devlin et al., 2019). Learned positional embeddings are more flexible in a way that position representation can adapt to a specific task during backpropagation. An alternative approach to accounting for the order in sequences is to inject relative positional encoding. Relative positional encoding encodes the position of a word in relation to other words in the sentence (Shaw et al., 2018). The motivation behind this is that self-attention could benefit more from pairwise positional relationships between input elements.

Since sequential position encoding is usually not suitable for processing graphs, (Kreuzer et al., 2021) define a Laplacian positional encoding for graphs. Their primary aim is to investigate how eigenfunctions of the Laplacian can be used to define absolute and relative positional encodings in graphs. If we want to draw an analogy with a sine/cosine positional encoding in Eq. 3.1, in Euclidean space, the Laplacian operator corresponds to the gradient divergence and its eigenfunctions are sine/cosine functions, with the squared frequencies corresponding to the eigenvalues. Moreover, the authors introduce principles from spectral graph theory which should be considered when constructing PEs for graphs. This includes normalization, eigenvalues, multiplicities, variable number of eigenvectors, and sign invariance.

Feed-forward network

A feed-forward network is one of the first neural network architectures and the simplest type of neural network. It consists of an input layer of nodes, one of a couple of hidden layers, and an output layer. Hidden layers comprise nodes, weights, and activation functions. Weights are defined as $\Theta = \{W, B\}$, where W is a set of connections between different nodes, and B is a set of biases. In this type of neural

network, information moves only in one direction — from the input layer, through the hidden layers, and to the output layer. The sum of the products of weights and the inputs is calculated in each node. This linear combination is then passed to an activation function $\sigma(\cdot)$ to generate the output.

$$f(x) = \sigma(w^T x + b). \tag{3.2}$$

The typical activation function is the *sigmoid* function that takes a real-valued x and gives an output between 0 and 1:

$$f(x) = \frac{1}{1 + \exp(-x)}. \tag{3.3}$$

Hyperbolic tangent function is another common activation function that takes a real-valued number and outputs a value between -1 and 1:

$$than(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} \tag{3.4}$$

Both the hyperbolic tangent and sigmoid activation functions map large negative and positive inputs into $[-1; 1]$ or $[0, 1]$, which causes saturation in practice.

Another most used activation function right now is *Rectified Linear Unit (ReLU):*

$$f(x) = max(0, x). \tag{3.5}$$

The last common activation function is the *softmax* activation function which is used to convert the output to a probability distribution:

$$f(x)_k = \frac{\exp(x_i)}{\sum_{j=1}^{K} \exp(x_j)}, \tag{3.6}$$

where K is the number of classes in the multi-class classifier.

3.1.2 Attention mechanism

According to psychology, attention is the cognitive process of selectively concentrating on one or a few things while ignoring others. Similar to how neural networks attempt to mimic human brain actions in a simplified manner, the attention mechanism is also an attempt to implement the same action of selectively concentrating

on a few relevant things while ignoring others in deep neural networks.

The attention mechanism emerged as an improvement of the encoder-decoder-based neural machine translation system in Natural Language Processing (NLP). This mechanism, and its variations, was later used in other applications, including computer vision, speech processing, etc. (Bahdanau et al., 2014) first proposed the attention mechanism as a way of improving recurrent neural networks (RNNs) or Long Short-Term memory (LSTM), which were the most popular approaches for neural machine translation at that time. The authors claimed that a potential issue with a family of encoder-decoder approaches was that a neural network needed to be able to compress all the necessary information of a source sentence into a fixed-length vector. This made it difficult for such models to cope with long sentences, especially those that are longer than the sentences in the training set. The authors of (Cho et al., 2014) have shown that encoder-decoder models suffer from a significant drop in translation quality when translating long sentences. Even though LSTM is supposed to capture the long-term dependencies better than the RNN, it tends to become forgetful in specific cases. Another problem with LSTM or RNN is that there is no way to give importance to some of the inputs compared to others while translating the sentence. So, considering all these drawbacks of encoder-decoder models, the elegant idea of (Bahdanau et al., 2014) has made the most valuable breakthroughs in deep learning research.

In order to address the bottleneck problem that arises with the use of a fixed-length encoding vector, where the decoder would have limited access to the information provided by the input, the authors of (Bahdanau et al., 2014) propose a new encoder-decoder architecture, where an encoder is a bidirectional RNN and a decoder that imitates the process of searching through the original source sentence while generating a translation. Let $\mathbf{x} = (x_1, x_2, \ldots, x_T)$ be the input sequence and $\{y_1, \ldots, y_{t-1}\}$ be the previously predicted words. The authors (Bahdanau et al., 2014) propose a decoder that models the following conditional probability

$$p(y_i|y_1, \ldots, y_{i-1}, \mathbf{x}) = g(y_{i-1}, s_1, c_i), \tag{3.7}$$

where s_i is an RNN hidden state for time i computed as $s_i = f(s_{i-1}, y_{i-1}, c_i)$ and c_i is a context vector that depends on a sequence of annotations $(h_1, \ldots, h_T)$ to which the encoder maps the input sequence. The context vector is then computed as a

weighted sum of annotations:

$$c_i = \sum_{j=1}^{T} \alpha_{ij} h_j. \tag{3.8}$$

The weight α_{ij} of each annotation h_j is defined as $\alpha_{ij} = softmax(e_i j)$ with $e_{ij} = a(s_{i-q}, h_j)$ being an alignment model. In the original paper, a is parameterized as a feed-forward network.

The aforementioned formulation can be interpreted as an attention mechanism, which can be further generalized (Brauwers and Frasincar, 2021). The input of the general attention mechanism is the matrix of feature vectors $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{d_f \times m}$, the *query* $\mathbf{q} \in \mathbb{R}^{d_q}$, the *keys* matrix $\mathbf{K} = [\mathbf{k}_1, \ldots, \mathbf{k}_m] \in \mathbb{R}^{d_k \times m}$, and the *values* matrix $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_m] \in \mathbb{R}^{d_v \times m}$. Comparing these three components to the attention mechanism discussed above the query would be analogous to the previous decoder output s_{t-1}, and the values would be analogous to the encoder inputs h_i. The keys and values are the same vectors in (Bahdanau et al., 2014). The general way to obtain matrices $\mathbf{K}$, $\mathbf{V}$ and $\mathbf{Q}$ is via a linear transformation of $\mathbf{F}$ using some weight matrices $\mathbf{W}_K \in \mathbb{R}^{d_k \times d_f}$ and $\mathbf{W}_V \in \mathbb{R}^{d_v \times d_f}$:

$$\mathbf{K} = \mathbf{W}_K \mathbf{X} \text{ and } \mathbf{V} = \mathbf{W}_V \mathbf{X} \tag{3.9}$$

The weight matrices are usually learned during the training. They can be initialized in different ways such as the identity matrix to retain the original feature vectors or using completely separate inputs for the keys and values (Brauwers and Frasincar, 2021).

The query $\mathbf{q}$ and the keys matrix $\mathbf{K}$ are used to calculate the attention score vector $\mathbf{e} = [e_1, \ldots, e_m]$:

$$e_i = score(\mathbf{q}, \mathbf{k}_i) \tag{3.10}$$

with $e_i \in \mathbb{R}^{1 \times 1}$, $\mathbf{q} \in \mathbb{R}^{d_q \times 1}$ and $\mathbf{k}_i \in \mathbb{R}^{d_k \times 1}$. The different types of existing *score* functions and their advantages are discussed further in the section. Moreover, the attention score e_i indicates how important the information contained in the key vector $\mathbf{k}_i$ is according to the query and the query vector serves as a request for information (Brauwers and Frasincar, 2021).

After we have calculated the attention sore, we proceed further to an alignment layer. The alignment layer is used to normalize the attention score which can gener-

ally have a wide range outside of $[0, 1]$:

$$a_i = align(e_i; \mathbf{e}) \tag{3.11}$$

with $\mathbf{e} \in \mathbb{R}^{m \times 1}$. A variety of the *align* functions is also discussed further in this section. The intuition behind the alignment layer is how important each feature vector is relative to the others for the particular problem.

Finally, the vector of attention weights allows us to compute a context vector $\mathbf{c} \in \mathbb{R}^{d_v \times 1}$:

$$\mathbf{c} = \sum_{i=1}^{m} a_i \mathbf{v}_i, \tag{3.12}$$

where $\mathbf{v}_i \in \mathbb{R}^{d_v \times 1}$. The context vector is then used in the further layers of the model. For instance, it can be translated directly into an output prediction with a softmax layer.

Due to the big success of the attention mechanism in machine translation, it has become extended into other fields of deep learning such as computer vision and graph representation learning. In order to adapt the attention mechanism to different problems, researchers have been exploring various forms of score functions, which have their own advantages and disadvantages. We begin the discussion of the most used attention score functions.

Attention scoring

The attention score function in Eq.3.10 is an important component of the attention mechanism. The most popular choices for the score function are the multiplicative (dot-product) score function (Luong et al., 2015) in Eq. 3.13 and the additive score function (Bahdanau et al., 2014) in Eq. 3.14 due to their simplicity.

$$\mathbf{e} = q^T k_i, \tag{3.13}$$

$$\mathbf{e} = \mathbf{w}^T \sigma(\mathbf{W}_1 \mathbf{q} + \mathbf{W}_2 \mathbf{k}_i) \tag{3.14}$$

where σ is an activation function, $\mathbf{w} \in \mathbb{R}^{d_w}$, $\mathbf{W}_1 \in \mathbb{R}^{d_w \times d_q}$, and $\mathbf{W}_2 \in \mathbb{R}^{d_w \times d_k}$ are trainable parameters.

One of the advantages of the multiplicative score function is computational efficiency due to highly optimized matrix operations. However, when the dimension

of the key matrix $\mathbf{K}$, d_k is too large, the vector $\mathbf{e}$ in Eq. 3.10 becomes also large in magnitude. When the softmax function in the alignment step is applied, the gradient will become very small which results in the vanishing gradient problem. To tackle this problem, (Vaswani et al., 2017) proposed to scale the dot-product attention by the factor $\frac{1}{\sqrt{d_k}}$.

The choice of a scoring function is most often based on empirical experiments. However, if computational efficiency is more important, then the dot-product function and scaled dot-product function is typically the best choice. For instance, (Vaswani et al., 2017) uses the scaled dot-product function since the Transformer is generally computationally expensive.

Attention alignment

The alignment layer in Eq. 3.11 determines to which parts of the input data the model will attend. The most popular function for the alignment layer is a simple softmax function, which is also used by (Vaswani et al., 2017) in the Transformer:

$$a_l = \frac{\exp e_l}{\sum_{j=1}^{m} \exp e_j}, \tag{3.15}$$

where $\mathbf{e}$ is the attention score vector calculated in the previous step.

Due to the nature of the softmax function, every part of the input receives some amount of attention (Brauwers and Frasincar, 2021). Moreover, the probabilistic interpretation of the softmax function makes it easier to understand which parts of the input are important to the output prediction. This alignment method is often referred to as *deterministic soft alignment* (Xu et al., 2015). The interpretation of the soft alignment can be seen as the model attending to all feature vectors. *Stochastic hard alignment* (Xu et al., 2015) tries to archive a more focused form of the alignment and can be interpreted as the model attending to one feature vector. Instead of using all weights $a_1, \ldots, a_{n_f}$ for computing a context vector in Eq. 3.12, we consider a value $m \in \mathbb{R}^1$ which is drawn from a multinomial distribution parametrized by $\{a_i\}$. Then, the context vector is computed as $c = \mathbf{v}_m$, where $\mathbf{v}$ is again a values vector. The main drawback of the stochastic hard alignment is that it is non-differentiable which means that it is impossible to use the standard gradient decent for the model optimization. (Xu et al., 2015) train a model with stochastic hard attention by maximizing an approximate variational lower bound or by using a reinforcement learning technique REINFORCE (Williams, 1992).

The other two types of the alignment layer are *local alignment* (Luong et al., 2015) and *reinforced alignment* (Shen et al., 2018). They are a compromise between hard and soft alignment. Local alignment implements a softmax distribution, similar to soft alignment. However, the softmax distribution is computed based only on a subset of the input. To determine the subset of the input, the authors use a window placed on the input. Reinforced alignment computes the soft alignment on a subset of the feature vector. But unlike local alignment, reinforced alignment uses a reinforcement learning agent, similar to hard alignment, to choose a subset of feature vectors. Soft alignment is typically regarded as the standard alignment function in attention models.

In the original Transformer (Vaswani et al., 2017), the multi-head attention mechanism can be written as follows

$$MultiHead(Q, K, V) = Concat(h_1, \ldots, h_H)W^O, \tag{3.16}$$

with

$$h_i = a(QW_i^Q, KW_i^K, VW_i^V) \tag{3.17}$$

$$a(Q, K, V) = softmax\Big(\frac{QK^T}{\sqrt{(d_k)}}\Big)V. \tag{3.18}$$

3.1.3 Variations of the Transformer

A wide variety of models have been proposed based on the original Transformer which can be divided into three groups: types of architecture modification, pre-training methods, and applications. A more detailed survey on various transformers can be found in (Lin et al., 2022). We will highlight several works that we find interesting and have some connections to our work. We first discuss the adaptation of Transformer architecture to graph-structure data. Even though we intended to use the original Transformer architecture, one can claim that we use Graph Transformer because we also leverage graph connectivity and update an attention matrix based on the local neighbourhood of each node. Another set of methods will concern the computational complexity of the vanilla Transformer. They try to reduce the complexity of the attention mechanism. In the case of our application, in order to observe metastable behaviour it is necessary to have many time points of a time-evolving graph, which might increase the training time of our method. Therefore, it

is worth mentioning these variations of the Transformer.

Graph Transformer

Since the original Transformer was designed for natural language processing purposes, it operates on all connections between words in a sequence. Such architecture does not leverage graph connectivity, and it can perform poorly when the graph structure is important and has not been used during the model training. Therefore, several works have been proposed to adopt the transformer to graph-structure data. (Dwivedi and Bresson, 2020) introduce the Graph Transformer Layer and Graph Transformer Layer with edge features. They claim that there are two main aspects in the development of a Graph Transformer which are sparsity and positional encoding. They suggest utilizing Laplacian eigenvectors as a positional encoding. They closely follow the original transformer architecture, however, the node update is defined in such a way so that the graph topology is taken into account:

$$h_i^{l+1} = O_h^l \|_{k=1}^{H} \Big(\sum_{j \in \mathcal{N}_i} w_{ij}^{k,l} V^{k,l} h_j^l \Big), \tag{3.19}$$

where

$$w_{ij}^{k,l} = softmax_i \Big(\frac{Q^{k,l} h_i^l K^{k,l} h_j^l}{\sqrt{d_k}} \Big), \tag{3.20}$$

and $Q^{k,l}, K^{k,l}, V^{k,l}$ are the keys, values and queries matrices, H denotes the number of attention heads, and $\|$ is a concatenation operation.

Another work (Li et al., 2019) suggests using the attention mechanism to all nodes in the graphs instead of a local neighbourhood of a specific node with the purpose of capturing global information in the graph. (Zhang et al., 2020b) introduce Graph-BERT for graph representation learning. They propose to use batches of linkless subgraphs sampled from the original graph.

Switch Transformer

Switch Transformer (Fedus et al., 2022) is another variant of the Transformer and one of the first trillion-parameter models. They use a modified version of a mixture-of-experts (MoE) paradigm that was developed by (Jacobs et al., 1991). The idea is that instead of activating multiple experts and combining their output, Switch Routing chooses a single expert to handle a given input. This allows training a model

on different GPUs while reducing communication costs. In order to implement it, they replaced the standard FFN layer in the original transformer with a switch feed-forward neural network. The illustration of a Switch Transformer encoder block is depicted in 3.1(b). When each token passes through this layer, it first passes through a router function which then routes the token to a specific expert. This results in high sparsity of the model meaning that not every parameter is utilized for every token.

Moreover, in order to take advantage of data- and model-parallelism, the Switch Transformer utilizes the newly introduced Mesh-TensorFlow (Shazeer et al., 2018). It is a language for specifying a general class of distributed tensor computations. In the Mesh-TensorFlow, a user can specify any tensor dimensions to be split across any dimensions of a multi-dimensional mesh of processors. The authors of the Switch Transformer experimented with different dimensions and they found that the most efficient dimension for scaling was the number of experts.

Scaling the attention mechanism

There are several attempts made to increase the computation efficiency of the attention mechanism. The complexity of the vanilla Transformer increases quadratically with the number of tokens in the input sequence and this is due to the estimation of regular (softmax) full-rank attention. For example, *Linformer* (Wang et al., 2020b) proposes to approximate the stochastic matrix formed by self-attention with a low-rank matrix. They decompose the original scaled dot-product attention into multiple smaller attentions through linear projections. In this way, the combination of these operations results in a low-rank factorization of the original attention.

Routing Transformer (Roy et al., 2021), in turn, relies on sparsity of the attention. It introduces the idea that was also used in Switch Transformer. The idea lies in the clustering of attention. Each attention module considers a clustering of the space: the current time step only attends to context belonging to the same cluster.

Another approach is called *Performers* (Choromanski et al., 2020). Instead of introducing sparsity in the attention or using low-rankness, they accurately estimate (softmax) full-rank attention of only linear space and time complexity. They propose a new method called fast attention via the positive orthogonal random features mechanism. This method consists of several parts. The first part, the FA-part, represents queries and keys as matrices of the form $\mathbf{A}(i,j) = \mathbf{K}(\mathbf{q}_i, \mathbf{k}_i)$, where q_i and

k_i represent the ith and jth query/key row-vector in Q and K respectively. Then, a kernel k $\mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_+$ with a feature map $\phi : \mathbb{R}^d \to \mathbb{R}^r_+$ as:

$$k(x, y) = \mathbb{E}[\phi(x)^T \phi(y)]. \tag{3.21}$$

We can now define new matrices of queries and keys Q', $K' \in \mathbb{R}^{L \times r}$ with rows given by $\phi(\mathbf{q}_i^T)^T$ and $\phi(\mathbf{k}_i^T)^T$. Finally, the original attention mechanism defined in Eq. 3.18 can be rewritten as:

$$Att(Q, K, V) = \hat{D}^{-1}(Q'((K')^T V)), \tag{3.22}$$

with

$$\hat{D} = diag(Q'((K')^T \mathbf{1}_L)). \tag{3.23}$$

Another part is the R-part which approximates the softmax kernel by using trigonometric functions that are then regularized using positive random features. The last part is the O-part which tries to entangle samples to be orthogonal using the Gram-Schmidt orthogonalization procedure. This is done to reduce the variance of the softmax of the estimator.

3.2 Contrastive representation learning

3.2.1 Problem formulation

Conventional supervised learning methods heavily rely on the availability of annotated training data. However, the scarcity of annotations has prompted researchers to explore alternative approaches that can effectively utilize unlabeled data. Contrastive learning addresses this problem by learning useful representations from unlabelled data which means that contrastive learning is a type of unsupervised learning technique. The goal of contrastive learning is to maximize the similarity between similar instances while minimizing the similarity between dissimilar instances.

Contrastive learning originated from the concept proposed in (Chopra et al., 2005). They introduced a method for training a similarity metric from data for face recognition and face verification applications. The idea was to find a function that maps input patterns into a target space such that a distance (e.g. the Euclidean distance) in the target space approximates the "semantic" distance in the input space. Formally, given a family of the function $F_\theta(x)$ parameterized by θ, the objective is

to find a value of the parameter θ such that the similarity metric $E_\theta(x_1, x_2) = \| F_\theta(x_1) - F_\theta(x_2) \|$ is small if x_1 and x_2 belongs to the same class, and large if they belong to different classes. The motivation was that methods that require known labels in advance are not suitable for applications where the number of classes is very large but the number of samples for each class is small.

The task of dimensionality reduction can also be seen as an application of contrastive learning. We need to map a set of high-dimensional input points into a low-dimensional manifold so that "similar" points in input space are mapped to nearby points on the manifold. Therefore, (Hadsell et al., 2006) used a concept of contrastive learning for solving a dimensionality reduction problem. They formulate the problem in the following way. Given a $D-$ dimensional vector, it aims to find a parametric function that transforms this vector into a $d-$dimensional vector, where $d \ll D$, while preserving as much information as possible. The authors place the following constraints on the function: (i) the distance measures in the output space should approximate the neighbourhood relationship in the input space; (ii) it should be able to learn invariances to complex transformations; (iii) it should remain accurate even for samples with unknown neighbourhood relationships. To find this parametric function, the authors propose to use contrastive loss where the input is pairs of samples $(\mathbf{x}_1, \mathbf{x}_2)$ with y being a binary label assigned to the pair. If $\mathbf{x}_1$ and $\mathbf{x}_2$ are similar, $y = 0$, otherwise $y = 1$. In order to define pairs of samples, it is suggested to use some prior knowledge or manual labelling etc. These pairs of similar or dissimilar samples are called *positive* and *negative pairs*.

Another formulation for contrastive representation learning can be viewed as learning by comparing. Thus, the goal is to find a low-dimensional space where samples from the same instance are pulled closer and samples from different instances are pushed apart. Formally, given a vector of input samples $x_i, i = 1, \ldots, N$ with corresponding labels $y_i \in \{1, \ldots, C\}$ with C being a number of classes, contrastive learning aims to learn a function $f_\theta(x)$ that encodes x_i into an embedding vector such that examples from the same class have similar representations and samples from different classes are far away from each other in a new space. However, it is worth mentioning that in contrastive learning representation, we do not have ground-truth labels or classes.

All above mentioned definitions of contrastive learning are facing a similar challenge — how to define positive and negative pairs if we know in advance that our data is not labelled. Therefore, since one of the key components of contrastive rep-

resentation learning is a contrastive loss, there have recently been proposed many contrastive losses that propose different ways of defining positive and negative pairs. We will further discuss some of the most popular contrastive losses and how they split the training data into negative and positive pairs.

3.2.2 Contrastive loss

Contrastive loss is one of the earliest training objectives used for deep metric learning in semi-supervised and unsupervised settings. In early versions of contrastive loss functions, only one positive and one negative sample are involved. These types of contrastive losses suffer from slow convergence (Sohn, 2016). However, there has been a recent shift in the trend, where many contrastive objectives now incorporate multiple positive and negative pairs within a single batch. We will focus on the latest state-of-the-art contrastive objectives as this field is rapidly advancing.

Triplet Loss

The most common approach to contrastive learning was proposed in (Ying et al., 2018b; Schroff et al., 2015) as a part of FaceNet, a deep convolutional network, that learns face recognition of the same person at different poses and angles. By using contrastive learning, the network is trained such that the squared L2 distances in the embedding space directly correspond to face similarity. The idea of the triplet loss is the following. Given one anchor input x, we select one positive sample x^+ and one negative x^-. This means that x and x^+ belong to the same class and x^- is sampled from another class than x. Triplet loss aims to minimize the distance between the anchor x and x^+ and maximize the distance between the anchor x and x^- at the same time in the following way:

$$\mathcal{L}(x, x^+, x^-) = \sum_{x \in \mathcal{X}} \max\left(0, \|f(x) - f(x^+)\|_2^2 - \|f(x) - f(x^-)\|_2^2 + \epsilon\right). \tag{3.24}$$

In order to improve contrastive learning, the negative sample should be challenging. Generating all possible triplets would result in many triplets that are easily satisfied, therefore, they would not contribute to the training. The authors proposed to use hard positive and hard negative samples by sampling them, within a mini-batch.

NCE

Noise Contrastive Estimation (NCE) is a method for estimating parameters of a statistical model, proposed by (Gutmann and Hyvärinen, 2010). The idea is to use logistic regression to separate the target data from noise. Given the target sample $x \sim P(x|C = 1, \theta) = p_\theta(x)$ and a noise sample $\tilde{x} \sim P(\tilde{x}|C = 0) = q(\tilde{x})$. The logit of a sample x from the target data distribution is modelled instead of the noise distribution using the logistic regression:

$$l_\theta(x) = \log \frac{p_\theta(x)}{q(x)} = \log p_\theta(x) - \log q(x). \tag{3.25}$$

The logits are then converted into probabilities with the sigmoid function and the cross-entropy loss is applied:

$$\mathcal{L}_{NCE} = -\frac{1}{N} \sum_{i=1}^{N} \Big[\log \sigma(l_\theta(x_i)) + \log \Big(1 - \sigma(l_\theta(\tilde{x}_i))\Big) \Big], \tag{3.26}$$

where the sigmoid function $\sigma(\cdot)$ is defined as

$$\sigma(x) = \frac{1}{1 + \exp(-x)}. \tag{3.27}$$

In the original paper, the NCE loss works only with one positive and one noise sample. However, this is addressed in more recent works on contrastive learning by introducing the comparison with multiple negative or noise samples.

InfoNCE

Another contrastive loss, InfoNCE, has been proposed by (Oord et al., 2018) and inspired by NCE. It uses categorical cross-entropy loss to identify the positive sample among a set of unrelated noise samples. Given a set of N random samples containing one positive sample $p(x_{t+k}|c_t)$ and $N - 1$ negative samples from some distribution $p(x_{t+k})$, we optimize:

$$\mathcal{L}_{InfoNCE} = -\mathbb{E}\Big[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)} \Big], \tag{3.28}$$

where c_t is called a context vector and $f(x, c) \propto \frac{p(x|c)}{p(x)}$ which is the estimation of the density ratio and it has a connection with mutual information optimization.

The paper claims that when predicting future information we encode the target x (future) and context c (present) into compact distribution vector representations in a way that maximally preserves the mutual information of the original x as

$$I(x, c) = \sum_{x,c} p(x, c) \log \frac{p(x|c)}{p(x)}. \tag{3.29}$$

Other contrastive losses include Lifted Structured Loss (Oh Song et al., 2016), N-pair Loss (Sohn, 2016), which introduces the comparison with multiple negative pairs, and Soft-Nearest Neighbors Loss (Frosst et al., 2019).

3.2.3 Strategies to successful contrastive learning

Based on various empirical experiments, different studies have proposed strategies on how to improve contrastive learning.

Batch Size and longer training

(Chen et al., 2020) has shown that the large batch size and longer training are key ingredients in the success of many contrastive learning methods, especially when it relies on in-batch negatives. When the batch size is big enough, the contrastive loss can cover a diverse collection of negative samples which allows the model to learn meaningful representation to distinguish different examples.

Bigger models

It has been shown that contrastive learning improves significantly with increasing the depth and width of a model. For supervised models, this statement holds as well (He et al., 2016), however, unsupervised learning benefits more from the bigger models than supervised models as shown in (Chen et al., 2020).

Hard negative sampling.

The idea of hard negative sampling appears in the context of triple loss defined in Section 3.2.2. The performance of triplet loss is highly dependent on the triplet selection strategy. The problem with triplets is that for a large part of the optimization, most triplet candidates already have the anchor much closer to the positive than the negative. Hard negative sampling, however, refers to the sampling where

triplets include an anchor sample where the positive sample from the same class is less similar than the negative sample from a different class (Xuan et al., 2020). Several works proposed approaches to how to define hard negative pairs (Robinson et al., 2020; Tabassum et al., 2022).

Other strategies for improving contrastive representation learning include memory banks of features from previous batches, online and offline mining (dynamically selecting hard negative samples during or before training), and adaptive weighting (assigning different weights to positive and negative pairs).

3.3 Novel unsupervised approach for embedding of time-evolving graphs with metastability

3.3.1 Problem formulation

Given a time-evolving graph $\mathcal{G}$ as a sequence of T graphs $\mathcal{G} = (G_0, \ldots, G_{T-1})$ at the consecutive time points $\{0, \ldots, T-1\}$ for some $T \in \mathbb{N}$. We assume that the time-evolving graph exhibits metastable behaviour and each metastable state is characterized by a certain structural pattern of the graph such as node clusters, edge clusters, or walks. Then, the problem can be divided into two sub-problems and they can be defined as follows: *Given a time-evolving graph $\mathcal{G} = (G_0, \ldots, G_{T-1})$ with assumed metastable behaviour, (i) we aim to find a mapping $f : \mathbb{G} \to \mathbb{S}$ that learns a low-dimensional representation of $\mathcal{G}$ such that the metastable properties in Def. 2.3.1 holds in a new lower dimensional space $\mathbb{S}$, (ii) and given a low dimensional space $\mathbb{S}$, we aim at finding a set of graph features $\mathcal{X} \in \mathbb{G}$ (node clusters, edge clusters, or walks) which are associated with the graph transitioning from one metastable state to another.* Unlike the method proposed in the previous chapter, the dimensionality of a new space $\mathbb{S}$ is a hyperparameter that has to be tuned in order to obtain a good performance.

The problem definition of this chapter is similar to the problem formulated in Section 2.3.1. However, here we are aiming to develop a method that can be used not only to learn a low-dimensional representation of a time-evolving graph but also we will be able to extract microbial biomarkers that are associated with transiting between different metastable states.

In the following sections, we will provide details about each component of the pro-

posed deep learning-based model. The chapter will finish by evaluating experiments and results.

3.3.2 Input

Let $\mathcal{G} = \{G_1, ..., G_T\}$ be a time-evolving graph with node features $\{x_t^v\}_{v \in V(G_t)}$, $t = 1, \ldots, T$. The input node features of each time-snapshot graph G_t are embedded to d_m-dimensional latent features via a linear projection and added to pre-computed node positional encodings. Moreover, each time-snapshot graph G_t has a corresponding adjacency matrix A_t. In order to capture the topological structure of a single time-snapshot graph G_t, we feed an adjacency matrix A_t to the Transformer as a mask, and we set attention weights to 0 whenever the corresponding adjacency matrix entries are 0. In such a way, we account for the neighbourhood properties of each time-snapshot graph.

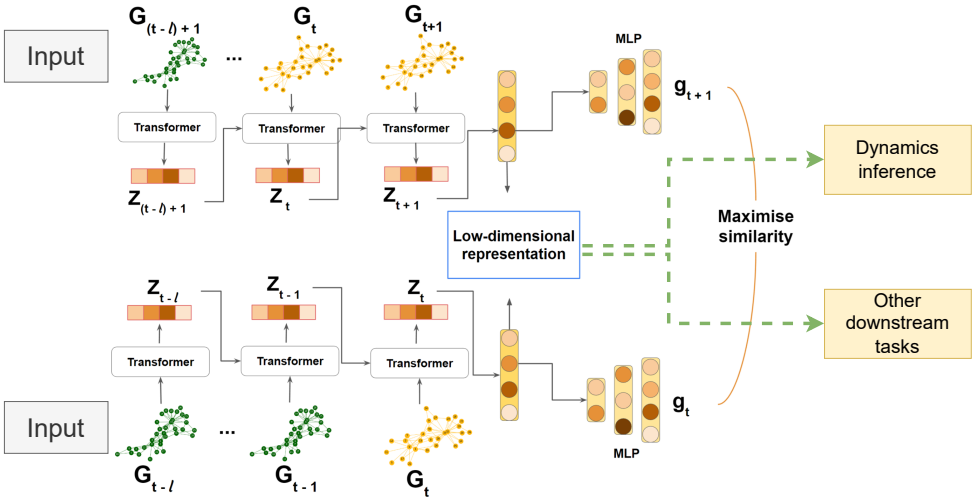


Figure 3.2: Overview of the method proposed in Chapter 3. The input to the model is two sets of consecutive time-snapshot graphs $\{G_{t-l}, \ldots G_t\}$ and $\{G_{(t-l)+1}, \ldots G_{t+1}\}$. For ah of this time-snapshot graph, we obtain a low-dimensional representation z_τ, which is mapped with a projection head where contrastive learning is implemented. The contrastive learning in the proposed method measures the similarity between two consecutive time-snapshot graphs.

3.3.3 Model architecture

In this section, we provide details about the model architecture and its components that play an essential role in learning a low-dimensional representation of a time-

evolving graph with metastability. The overall idea of our model can be found in Figure 3.2. The key component of our model is the Encoder of the Transformer, which has been discussed in Section 3.1.1 and contrastive learning discussed in Section 3.2. Further, we will explain how we learn a low-dimensional representation of a time-evolving graph while maintaining metastability in more detail.

Let $\mathcal{T} = \{t_k\}_{k=1}^{B}$ be a set of randomly sampled time points, and $\mathcal{G}_{\mathcal{T}} = \{G_{t_1}, \ldots, G_{t_B}\}$ be a mini-batch of time-snapshot graphs sampled from $\mathcal{G}$ with a mini-batch size B. The first key ingredient of the model is that we share the embedding of a time-snapshot graph with the consecutive time-snapshot graph in the temporal sequence to facilitate learning of temporal changes. We pass the resulting low-dimensional representation z_t of the entire G_t with the time-snapshot graph G_{t+1} as an additional input to the model. We control the number of historical representations with the hyperparameter l.

The next key component of our model is a master node. Similar to the natural language application, we defined a special node, which we call a master node. The master node is connected to all nodes in the time-snapshot graph. Initially, the master node is represented as a learnable, randomly initialized vector. (Gilmer et al., 2017) also incorporated a similar trick with a latent master node which is connected to every input node in the graph with a special edge type. It serves as a "station" to which each node reads from or writes on every step of message passing. The Transformer computes the embedding of the master node, which we regarded as a final low-dimensional representation of the time-evolving graph. This graph embedding is then passed as the initial master node to the consecutive time-snapshot graph in the temporal sequence. Moreover, since we connect the master node with all other nodes in each time-snapshot graph, the size of the adjacency matrix changes, $A_t \in \mathbb{R}^{(n+1)\times(n+1)}$. The reasoning behind this is that originally, the Encoder of the Transformer learns the embedding of nodes, we, however, aim at learning the embedding of the entire time-snapshots graphs. Therefore, we consider the embedding of the master node as an embedding of a corresponding time-snapshot graph. Formally, we update the graph embedding z_t of the time-snapshot graph G_t

recursively as follows:

$$\begin{aligned} z_1 &= R(z_0, x_1, A_1), \\ z_2 &= R(z_1, x_2, A_2), \\ &\dots \\ z_t &= R(z_{t-1}, x_t, A_t), \end{aligned} \tag{3.30}$$

where R is the Encoder of the Transformer that updates embedding of nodes, $z_t \in \mathbb{R}^{d_m}$ is a master node, x_t is a vector of node features and $A_t \in \mathbb{R}^{(n+1)\times(n+1)}$ denotes an adjacency matrix of G_t.

Finally, we project the graph embedding $z_t \in \mathbb{R}^{d_m}$ of the time-snapshot graph G_t into the space with the dimension d via a linear projection. This embedding will be considered a final low-dimensional representation of the time-evolving graph $\mathcal{G}$ that can further be used for downstream tasks. We denote the final embedding of the time-snapshot graph with $\hat{g}_t \in \mathbb{R}^d$:

$$\hat{\mathbf{g}}_t = W z_t + b,$$

where W and b are learnable parameters.

We use two hidden layers and a non-linear activation function in order to project the mini-batch of resulting low-dimensional vector representations $\hat{\mathbf{g}}_{t_k}$ and $\hat{\mathbf{g}}_{t_k+1}$ into the space where contrastive learning is conducted, as it is done in (Chen et al., 2020):

$$\mathbf{g}_{t_k} = W^{(2)}\sigma(W^{(1)}\hat{g}_{t_k}) \tag{3.31}$$

where $W^{(1)}$, $W^{(2)}$ are learnable parameters, $\hat{\mathbf{g}}_{\mathbf{t_k}} \in \mathbb{R}^d$, $\mathbf{g}_{\mathbf{t_k}} \in \mathbb{R}^{\frac{d_m}{2}}$ and σ is an activation function. The projected vectors $\mathbf{g}_{t_k}$ and $\mathbf{g}_{t_k+1}$ are used to optimize a contrastive loss. Algorithm 1 summarizes the proposed method.

Furthermore, we explain how we use contrastive learning to make embeddings of consecutive time-snapshots graphs to preserve the metastable behaviour in the low-dimensional space.

3.3.4 The choice of contrastive loss

In the previous section, we have highlighted the intuitive and formal ideas of contrastive learning and state-of-the-art contrastive objectives that work with one positive

Algorithm 1: Main learning algorithm

Input:

l: the number of historical representations;

$\mathcal{G} = \{G_1, \ldots, G_T\}$: a time-evolving graph such that each time-snapshot graph $G_t = (x_t, A_t)$;

$\mathbf{z}_{init}$: a randomly initialized learnable master node; R: the Transformer;

$W, W^{(1)}, W^{(2)}, b$: parameters of the model;

$\mathcal{L} = 0$;

for *sampled mini batch of indices* $\{t_k\}_{k=1}^{B}$ **do**

$\mathbf{z}_{curr} = \mathbf{z}_{init}$; $\mathbf{z}_{next} = \mathbf{z}_{init}$

for $j = 1$ **to** l **do**

#time-snapshot graphs at time *t*

Select time-snapshot graphs $\{G_{t_k+j}\}_{k=1}^{B}$

$\mathbf{z}_{curr} = R(\mathbf{z}_{curr}, x_{t_k+j}, A_{t_k+j})$

#time-snapshot graphs at time *t+1*

Select time-snapshot graphs $\{G_{(t_k+1)+j}\}_{k=1}^{B}$

$\mathbf{z}_{next} = R(\mathbf{z}_{next}, x_{(t_k+1)+j}, A_{(t_k+1)+j})$

#final graph embeddings

$\hat{\mathbf{g}}_{curr} = W\mathbf{z}_{curr} + b$

$\hat{\mathbf{g}}_{next} = W\mathbf{z}_{next} + b$

#projection head for contrastive learning

$\mathbf{g}_{curr} = W^{(2)}\sigma(W^{(1)}\hat{\mathbf{g}}_{curr})$

$\mathbf{g}_{next} = W^{(2)}\sigma(W^{(1)}\hat{\mathbf{g}}_{next})$

#pairwise similarity

$sim = \frac{\mathbf{g}_{curr}^{T} \cdot \mathbf{g}_{next}}{\|\mathbf{g}_{curr}\| \cdot \|\mathbf{g}_{next}\|}$

$\mathcal{L} = \mathcal{L} + \sum_{i=1}^{B} -\log\left(\frac{\exp\left(sim_{i,i}/\tau\right)}{\sum_{j=1}^{B}\exp\left(sim_{i,j}/\tau\right)}\right)$

end

end

return $\mathcal{L}$

and one negative as well as multiple positive and negative samples. Here, we will explain how to apply contrastive learning for time-evolving graphs with metastability.

At the beginning of Chapter 3.2, we discovered that the first application of contrastive learning was in dimensionality reduction. Since our goal is to obtain the low-dimensional representation, or embedding, of the time-evolving graph, while preserving as much information about microbiome dynamics as possible, we will apply contrastive learning to our problem as well. However, we will present a new way of defining positive and negative pairs necessary for contrastive learning based on the following assumption.

Assumption 3.3.1. *According to the definition of metastability (2.3.1), the probability of two consecutive time-snapshot graphs G_t and G_{t+1} being similar is close to 1 and so should be the probability for their graph embeddings $\hat{g}_t$ and $\hat{g}_{t+1}$.*

In other words, we consider a pair of low-dimensional representations $(\hat{g}_{t_k}, \hat{g}_{t_k+1})$ as a positive pair and pairs $(\hat{g}_{t_k}, \hat{g}_{t_k+\tau}), \tau = 2, \ldots, N$ as negative pairs, where τ is randomly sampled. It is possible for negative samples to be of the same metastable states but at different time points. Finally, given low-dimensional vector representations $\mathbf{g_{t_k}}$ and $\mathbf{g_{t_k+1}}$ as the output of our model, we define the following loss function. First, we compute the similarity between $\mathbf{g_{t_k}}$ and $\mathbf{g_{t_k+1}}$

$$sim(\mathbf{g_{t_k}}, \mathbf{g_{t_k+1}}) = \frac{\mathbf{g_{t_k}}^T \cdot \mathbf{g_{t_k+1}}}{\| \mathbf{g_{t_k}} \| \cdot \| \mathbf{g_{t_k+1}} \|}. \tag{3.32}$$

With the above assumption, diagonal elements of $sim(\mathbf{g_{t_k}}, \mathbf{g_{t_k+1}})$ represent positive pairs and off-diagonal elements represent negative pairs. Then, the similarity score is fed to the NCE loss function:

$$\mathcal{L} = \sum_{i=1}^{N} -\log\Big(\frac{\exp\Big(sim(\mathbf{g_{t_k}}, \mathbf{g_{t_k+1}})_{i,i}/\tau\Big)}{\sum_{k=0,k\neq i}^{N} \exp\Big(sim(\mathbf{g_{t_k}}, \mathbf{g_{t_k+1}})_{i,k}/\tau\Big)}\Big). \tag{3.33}$$

where τ is a temperature hyperparameter. The experiments from (Wang and Liu, 2021) have shown that the temperature hyperparameter is important in controlling the local separation and global uniformity of the embedding distribution. Minimizing the loss function in Eq. 3.33 forces the parameters of the model to be tuned such that graph embeddings of two consecutive time-snapshot graphs appear to be as close as possible.

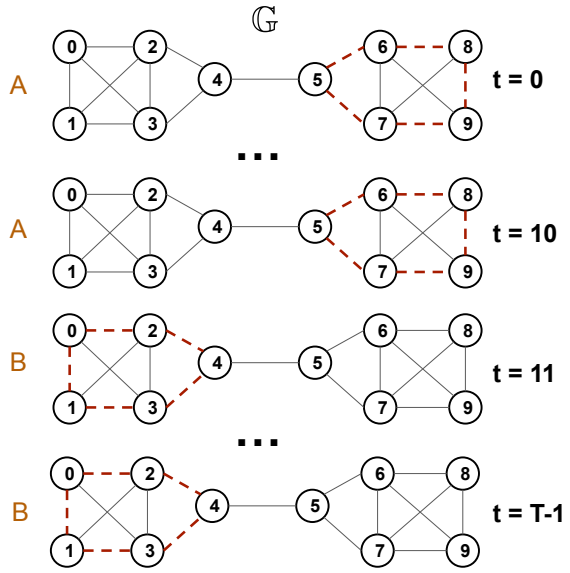


Figure 3.3: The example of a time-evolving graph with metastability whose two states A and B are difficult to distinguish since they are topologically the same. Red dashed edges are removed from the time-evolving graph $\mathcal{G}$.

We have presented the method, which is based on the Transformer and contrastive learning, to learn the low-dimensional representation of the time-evolving graph. Through a set of various experiments in the next section, we demonstrate on synthetic and real-world datasets that our method is capable of learning the low-dimensional representation of the time-evolving graph. The second part of the problem formulation defined in Section 3.3.1 will be explored in a separate chapter. The implementation of the proposed method can be found in https://github.com/k-melnyk/deep-metastability.git.

3.3.5 Positional encoding

In Section 3.1.1 we have discussed the importance of positional encoding in the transformer and different positional encodings for sequence data. We also highlighted that positional encoding used in sequence data is not suitable for processing graphs. Despite this fact, we will utilize the positional encoding defined in Eq. 3.1. Even though some papers have shown that using the graph Laplacian eigenvectors as a positional encoding is promising for graph-structure data, we were not able to demonstrate the effectiveness of the Laplacian positional encoding for our problem.

However, we would like to highlight the importance of positional information for

Table 3.1: Statistics of each dataset used in the Chapter 3.

Name	**#Nodes**	**#Edges**	**#Time steps**	**#States**
npos_2WellGraph	150	10821	10000	2
pos_2WellGraph	100	4109	10000	2
pos_3WellGraph	150	10869	10000	3
CholeraInf	96	106	34	2
MovingPic	919	10602	658	2

our problem. In order to illustrate this, we will demonstrate a straightforward toy example (Melnyk et al., 2023). Figure 3.3 presents a scenario in which different state structural characteristics are challenging to differentiate, posing difficulty for the method in discerning between metastable states within the evolving graph. We observe two states, denoted as A and B, within the time-evolving graph $\mathcal{G}$. These states are characterized by the removal of edges in different regions of $\mathcal{G}$. Despite the topological similarity between the states, leading to the same points in the low-dimensional space, distinguishing their structural characteristics proves challenging for our method. The same challenge applies to our synthetic dataset presented in Section 2.4.1, where the nodes within the circles represent metastable states, and the neighbourhoods of these nodes exhibit almost identical characteristics for the model. Therefore, we suggest that using positional encoding for the proposed model to understand the differences between structural graph features associated with different metastable states is crucial. We will extend the synthetic data with a case where we will have well-defined graph structural features. We will show in our experiments that in the case when we have clear separable graph features associated with metastable states, the model will perform well without introducing positional encoding to the model.

3.3.6 Training and evaluation

We divide each dataset into training and validation sets where each of the sets has a number of consecutive time-snapshot graphs. For each dataset, the node feature vectors are defined as a number of neighbours of nodes. The training is based on mini-batch gradient descent using the Adam optimization algorithm with the default parameters. The model was implemented in Python 3.10 using PyTorch.

In order to evaluate the proposed method in its ability to learn a low-dimensional

representation of a time-evolving graph while maintaining metastability, we will follow the same steps as in Section 2.3.3. However, we chose the best model based on the best ARI score during the training.

3.4 Synthetic data and Results

3.4.1 Synthetic data

In Section 2.4.1 we introduced the synthetic data used throughout this thesis. The idea was based on using a trajectory generated from a stochastic process with a certain potential function and removing edges between nodes that are falling in the corresponding circle at random. We will call this data positional synthetic data since it requires positional encoding as we discussed in the previous section. As we already said, we will introduce another synthetic data which is based on the same idea but this time we remove edges not randomly. This particular data type exhibits a distinct structural pattern compared to positional data. Instead of randomly removing edges between nodes within the circular region during the third step, we remove edges in a manner that ensures each node has a specific number of neighbours. The number of removed neighbours for each node is defined arbitrarily and varies across different states. We will call this data non-positional as it does not require a positional encoding in the model.

Using this idea of generating synthetic data, we will generate two positional datasets — pos_2WellGraph and pos_3WellGraph, and we will generate one non-positional data — npos_2WellGraph. The data statistics can be found in Table 3.1.

3.4.2 Experiments and Results

In this part of the thesis, we will evaluate the proposed method on a synthetic time-evolving graph with a dynamical property such as metastability. This chapter focuses on the first sub-problem defined in Section 3.3.1. Namely, we want to learn a time-evolving graph in a such way so that we can analyze microbiome dynamics in a low-dimensional space. As it was stated above, it is infeasible to extract the number of metastable states and their location using the proposed method, unlike the method from Chapter 2.

This section will be divided into similar paragraphs as in the previous chapter. We

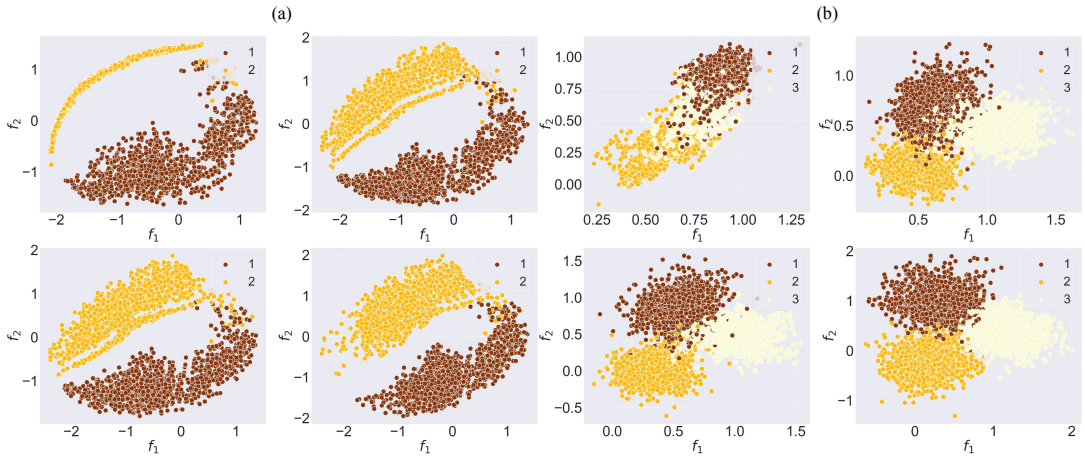


Figure 3.4: The evolution of the graph embedding of the time-evolving graph $\mathcal{G}$ during the training of the proposed model on **(a)** npos_2WellGraph and (**b**) pos_3WellGraph. The points are colour-coded according to ground-truth labels.

start with the analysis of how well the proposed method can learn graph embedding while maintaining metastable states. Then, we will compare our method with some other methods for the analysis of graph-structure data. We also demonstrate the impact of a temperature parameter of the contrastive loss in Eq. 3.33.

Experimental setup

For the analysis of metastability, we chose the number of embedding dimensions to be two. The hyperparameter of the history length l is set to be three. We train the model during 200 epochs. We use the Adam optimizer with the default parameters and a batch size of 64. By trial and error in architecture selection, the number of heads in the Transformer is four and the number of layers $N = 3$. We have conducted an additional analysis on choosing the temperature parameter which will be discussed further in this thesis.

In the comparative analysis, we compare the performance of our model with – node2vec, PCA, graph2vec, and Weisfeiler-Lehman kernel. For all models including the model proposed in this chapter, we set the number of dimensions of graph embedding to 32. We use the original implementations of node2vec and graph2vec with the default hyperparameters. Since node2vec is developed to learn node representations, we average node embeddings of each time-snapshot graph to obtain embeddings of the entire time-snapshots graph. Moreover, in order to evaluate how well the model

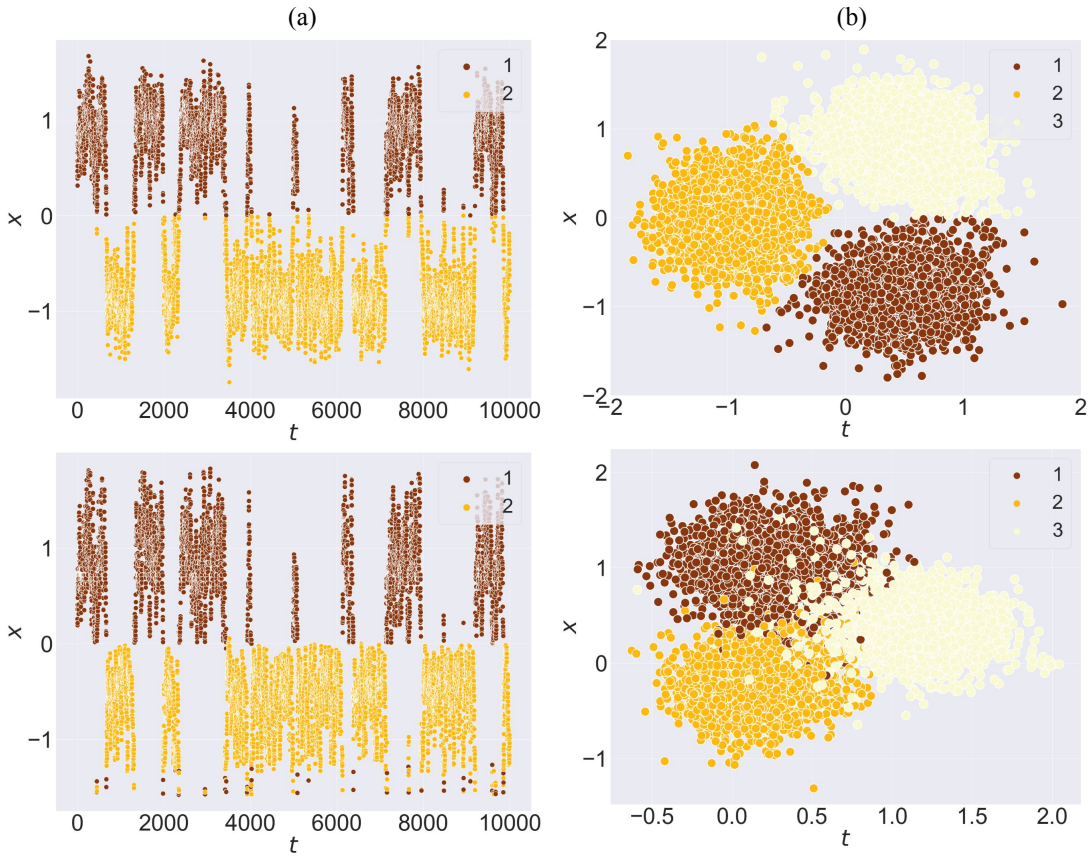


Figure 3.5: The comparison of an initial trajectory sampled from the SDE (Eq. 2.31) with a corresponding potential function (**top**) and the final graph embedding (**bottom**) for: **(a)** the *npos_2WellGraph* dataset and **(b)** the *pos_3WellGraph* dataset.

can capture metastable behaviour, we cluster points of resulting graph embeddings with k-means clustering method.

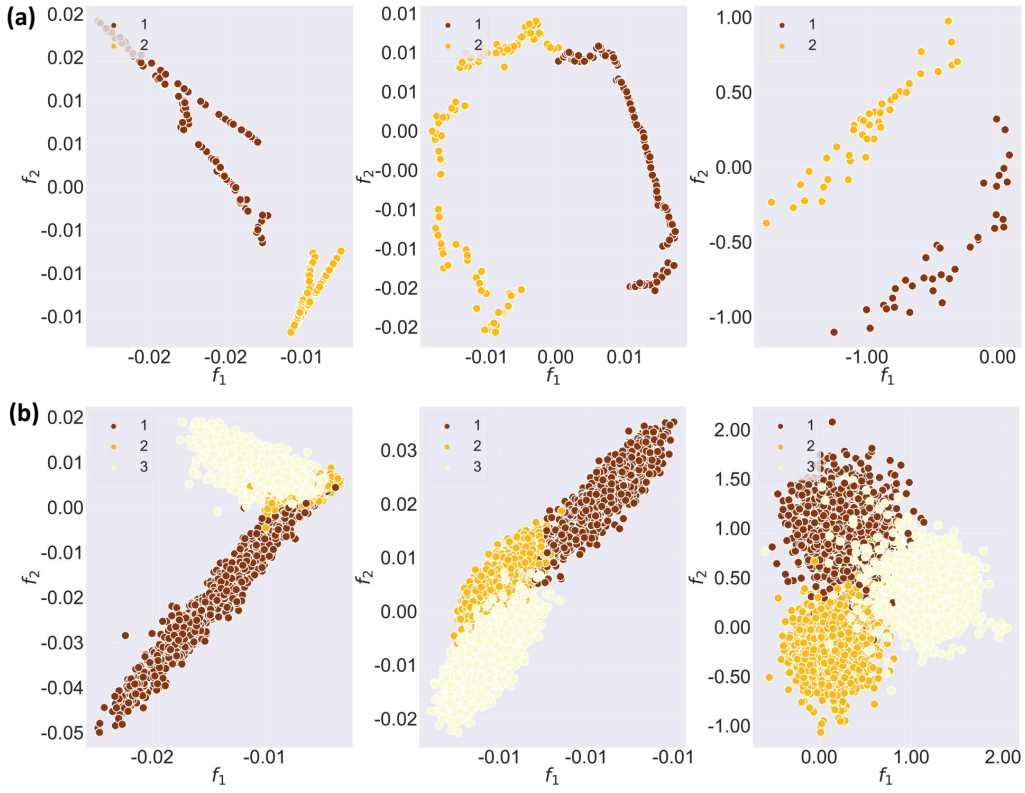


Figure 3.6: The graph embeddings of the time-evolving graph $\mathcal{G}$ for **(a)** npos_2WellGraph and **(b)** pos_3WellGraph. From left to right: PCA on adjacency matrices, PCA on eigenfunctions of graphKKE, and the proposed method from this Chapter.

Temperature parameter

To understand the importance of the temperature τ in the contrastive loss, we train the model with different temperature values on npos_2WellGraph. The values between 0.05 and 1.0 have been chosen. According to (Wang and Liu, 2021), the model with a small temperature tends to generate a more uniform distribution of graph embeddings and be less tolerant to similar samples. In our case, we have not noticed any dramatic changes in the performance. The ARI score for different temperature values can be seen in Figure 3.7. We can see that when the temperature parameter has a value of 0.5, the ARI on the training set decreases but the ARI on the test set has not changed dramatically in comparison to other temperature values.

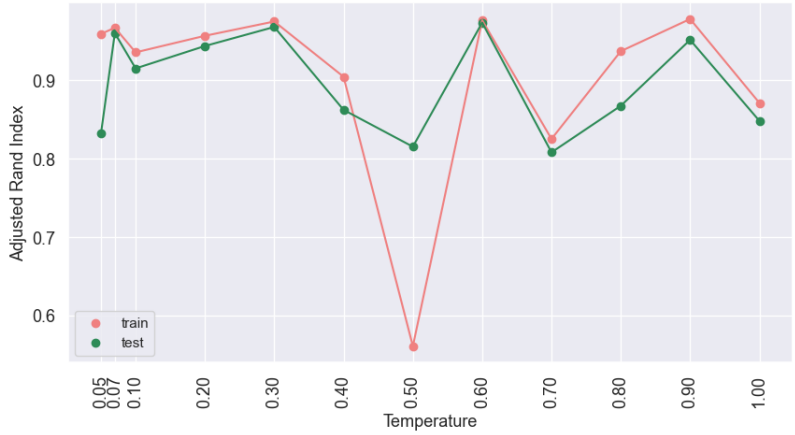


Figure 3.7: Adjusted Rand Index on the model trained on *npos_2WellGraph* with different temperature values of the contrastive loss. The red line corresponds to the ARIs on a training set and the green line indicates the ARIs on a test set.

Analysis of metastability

We will conduct the analysis of metastability in three steps. The first step is that we will plot a state of a low-dimensional representation of graph embedding at different epochs and colour-code the points based on the ground truth labels. The second step will include the comparison of a low-dimensional vector obtained by our model with the results of several other models. The first method will be PCA applied directly to adjacency matrices and the second method will be PCA applied to eigenfunctions obtained with graphKKE, a method proposed in Chapter 2. The third step is based on the comparison with the initial trajectories generated from SDE 2.31 with corresponding potential functions. For this experiment, we will set $d_m = 1$ for npos_2WellGraph and $d_m = 2$ for pos_3WellGraph.

The evolution of the graph embedding during the training for both synthetic datasets — npos_2WellGraph and pos_3WellGraph — are illustrated in Figure 3.4. The visualization demonstrates that during the training, our model tends to capture the underlying metastable structure in the time-evolving graph. Moreover, at the end of the training, we see that our model learns the graph embedding maintaining the initial metastable dynamics with two clear clusters for npos_2WellGraph and three clusters for pos_3WellGraph. Since the initial SDE trajectory has points that are located on the boundary between two metastable states, we can see for two datasets that some points are overlapping.

Furthermore, we compare the initial SDE trajectory and the final graph em-

Table 3.2: Adjusted Rand Index (ARI) for the comparative analysis on the graph clustering task for *the synthetic datasets*. ARI close to 1 corresponds to greater accuracy in correctly identifying the ground truth states, and an ARI value close to 0 stands for random clustering. The results are presented for the synthetic data.

Dataset	graphKKE	WL kernel	graph2vec	node2vec	PCA	Model
npos_2WellGraph	0.99	0.98	0.90	0.00	0.95	0.96
pos_2WellGraph	0.97	0.97	0.05	0.00	0.94	0.82
pos_3WellGraph	0.93	0.11	0.00	0.00	0.36	0.80

bedding obtained from our model. The result for npos_2WellGraph is presented in Figure 3.5(a) which shows that two trajectories are almost identical. The same result can be observed for pos_3WellGraph in Figure 3.5(b). These results indicate that the model is capable of extracting the underlying metastable dynamics in the time-evolving graph.

Finally, the visualization of a time-evolving graph embedding obtained from different methods is presented in Figure 3.6. It can be seen that PCA + adjacency matrices and graphKKE + PCA fail to produce a low-dimensional representation of a time-evolving graph with clearly separated metastable states in the case of pos_3WellGraph. For npos_2WellGraph, PCA + adjacency matrices and our method output the best-separated clusters.

Comparative analysis

Here, we will present more quantitative results by comparing the ARI of our method with several other methods. Table 3.2 demonstrates the ARI score for three synthetic datasets obtained with graphKKE, WL kernel, graph2vec, node2vec, PCA and our model. From the table can also be seen that the graphKKE method outperforms our model in the case of the pos_2WellGraph and pos_3WellGraph datasets. However, if we aim to have a lower dimensionality of the graph embedding, then this method will fail to produce the same clustering accuracy. It is worth reminding that graphKKE has a big advantage compared to any other methods as it is able to output the number of metastable states in a time-evolving graph. With other methods, we need to apply k-means clustering without knowing the number of clusters in advance. Moreover, considering the results of other graph representation learning, node2vec fails completely to learn the graph embedding of the time-evolving graph and graph2vec performs poorly on all synthetic datasets except npos_2WellGraph.

It remains unclear whether graph2vec struggles to identify states in the positional data because states do not have unique topological patterns, or because this method is not meant to capture temporal changes.

The experiments and results have shown that our model is able to learn a low-dimensional representation that can further be used for the analysis of dynamics. In the next section, we will show the real application of our model to the microbiome data.

3.5 Microbiome data and Results

3.5.1 Microbiome datasets

We start the section by discussing real-world microbiome datasets that will be used in this Chapter. We will use two microbiome datasets presented in Section 2.5.1. The first dataset comes from (Caporaso et al., 2011) and we call it MovingPic. In this study, one male and one female were sampled daily at three body sites (gut, skin, and mouth) for 15 months and for 6 months, respectively. In order to obtain a time-evolving graph, we pre-process Operational Taxonomic Units (OTU) that contain the number of 16S rDNA marker gene sequences that are observed for each taxonomic unit in each sample. Let $D \in \mathbb{R}^{T \times p}$ be an OTU table, where T is the number of time points and p is the number of OTUs. As this data has no obvious perturbations such as antibiotics exposure or diseases, which could potentially create a metastable structure, an artificial noisy signal is added to the data. The Pearson correlation between two OTUs is computed and then the initial time-snapshot graph is constructed. In order to construct time-snapshot graphs at each time step, we use the OTU table to remove edges between nodes. If the OTU count for a particular node is zero, then the edge is removed between this node and its neighbouring nodes.

The second real-world dataset is called CholeraInf and has been introduced in a study about the recovery from Vibrio Cholera infection (Hsiao et al., 2014). In this study, faecal microbiota were collected from seven cholera patients from disease (state 1) through recovery (state 2) periods. Moreover, in our experiment, we use the microbiome of one patient, since the variation in the microbiome constitution among individuals can have an impact on the result of the model. The time-evolving graph is obtained in the same way as it has been done for the MovingPic dataset, and more details can again be found in Section 2.5.1.

3.5.2 Experiments and Results

Experimental setup

As in the case of simulated data, we will focus on the analysis of metastability and the comparative analysis. For the analysis of metastability, we chose the number of embedding dimensions to be two. The hyperparameter of the history length l is set to be three. Since the sizes of the real-world datasets are small, we train the model for 50 epochs with a batch size of 32 for MovingPic and 8 for CholeraInf. We use the Adam optimizer with the default parameters and a batch size of 64. By trial and error in architecture selection, the number of heads in the Transformer is four and the number of layers $N = 3$.

In the comparative analysis, we compare the performance of our model with the performance of the same model as we used in the case of synthetic data, namely PCA, graphKKE, WL kernel, node2vec, and graph2vec.

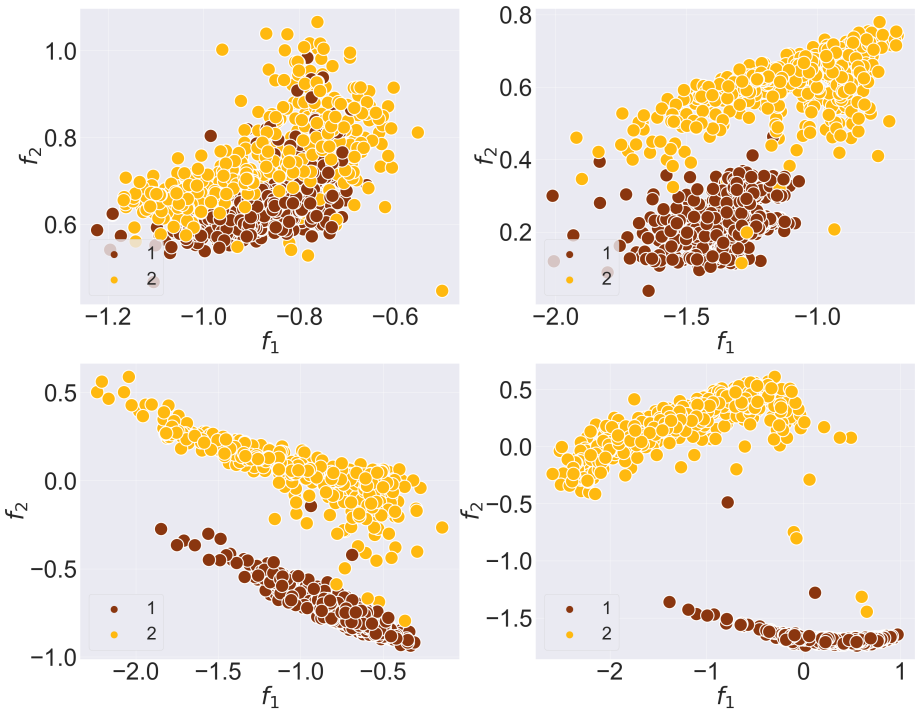


Figure 3.8: The evolution of the graph embedding of the time-evolving graph $\mathcal{G}$ during the training of the proposed model on *MovingPic*. The points are colour-coded according to ground-truth labels.

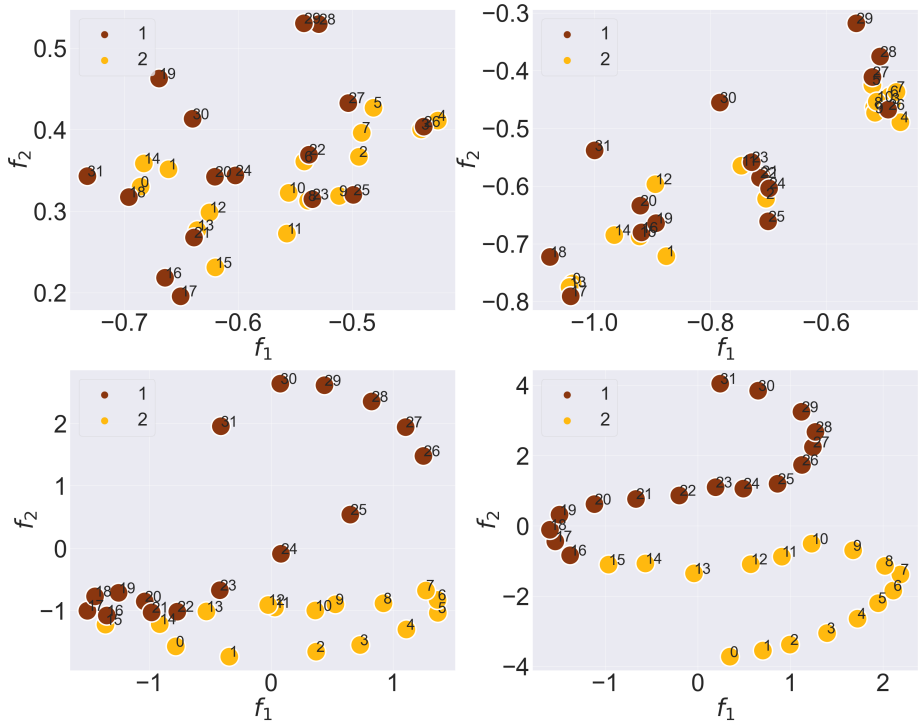


Figure 3.9: The evolution of the graph embedding of the time-evolving graph $\mathcal{G}$ during the training of the proposed model on *CholeraInf*. The points are colour-coded according to ground-truth labels.

Analysis of metastability

Similar to the analysis of metastability for synthetic data, we will split this part into exploring graph embedding during training and the comparison with other methods.

The evolution of the graph embedding during the training for MovingPic and CholeraInf is presented in Figure 3.8 and Figure 3.9, respectively. As it was in the case of synthetic datasets, our method is also able to identify the metastable behaviour in the time-evolving graph and preserve it in the new space. It can be seen that the low-dimensional representation of MovingPic has two clear-separated clusters, indicating the presence of two metastable states, the period of cholera infection and the period of recovery. For CholeraInf we have added time points from the original dataset to see if the low-dimensional space has the same time order as it was in the original, high-dimensional space. We can see on the bottom right plot that the points of the low-dimensional representation mimic the time order of the time-snapshot graphs. However, we have noticed that the order of points in a low-dimensional space changes when retraining the model which is an indication that the

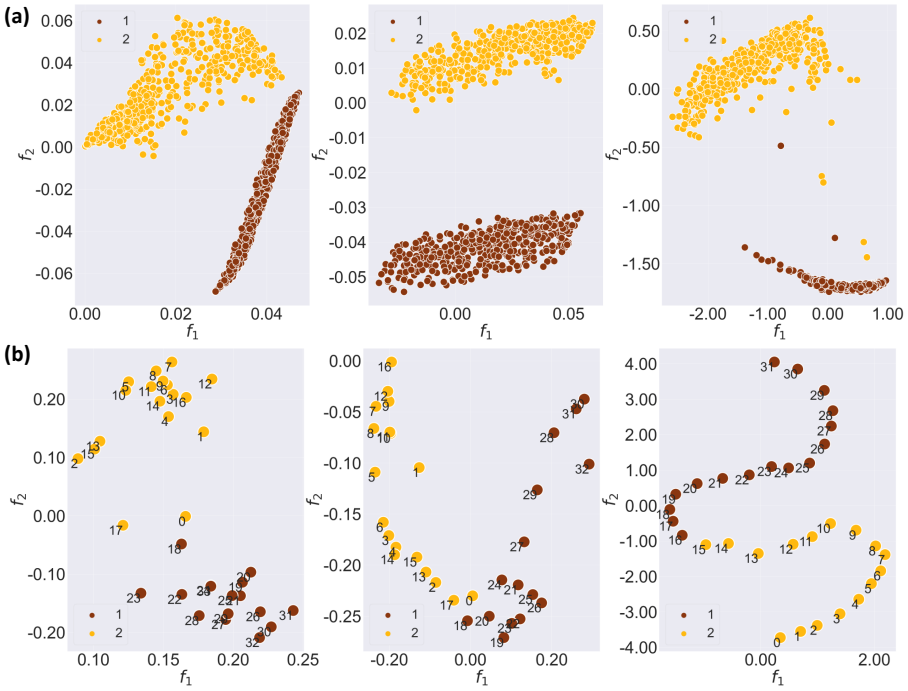


Figure 3.10: The graph embeddings of the time-evolving graph $\mathcal{G}$ for **(a)** MovingPic and **(b)** CholeraInf. From left to right: PCA on adjacency matrices, PCA on eigenfunctions of graphKKE and the result of the proposed method.

size of the dataset is too small. In this case, it is worth exploring transfer learning or fine-tuning.

If we compare graph embeddings from other methods with the graph embedding obtained from our model, it can be seen in Figure 3.10 that all methods give relatively similar low-dimensional representation. However, for the CholeraInf (Figure 3.10(b)) our model preserves consecutive time points in the new space which indicates that one metastable state (healthy) follows an alternative state (ill).

Comparative analysis

The second part of this experiment aims to compare our model with other dimensional reduction methods on the clustering task. From Table 3.3, it is evident that our model performs significantly better than WL kernel, graph2vec, and node2vec. Node2vec performs poorly across all real-world datasets similar to the synthetic datasets, which is the result of a lower-order substructure embedding method meaning that it can model only local similarities and fails to learn global topological similarities. Yet, graphKKE has a similar performance as the model proposed in this

Chapter. Interesting that the WL kernel performed as well as our model on the synthetic datasets, however, on the real-world datasets its performance is worse.

Table 3.3: Adjusted Rand Index (ARI) for the comparative analysis on the graph clustering task for *the real-world datasets*. ARI close to 1 corresponds to greater accuracy in correctly identifying the ground truth states and an ARI value close to 0 stands for random clustering.

Dataset	graphKKE	WL kernel	graph2vec	node2vec	PCA	Model
MovingPic	0.99	0.56	0.42	0.08	0.54	0.99
CholeraInf	0.88	0.65	0.29	-0.02	0.77	0.87

3.6 Discussion

This chapter aimed to further our study of learning a low-dimensional representation of a time-evolving graph with metastability. We addressed the challenges of the analysis of microbiome dynamics such as the complexity of interactions presented in the microbiome. We believe that representing the microbiome as a time-evolving network is crucial to understanding its dynamics and its impact on human health. This, however, requires approaches that will be able to deal with data that has temporal and structural aspects to be analyzed. Therefore, we proposed an unsupervised deep learning-based method to facilitate the analysis of the microbiome dynamics. The method aims at learning a low-dimensional representation of a time-evolving graph while maintaining dynamics. This representation can be further used to analyze the dynamics of the microbiome such as the probability of jumping from one state (healthy) to another (ill) or for further analysis with Transition Path Theory or Markov State Models to identify biologically relevant metastable sets or connecting paths between them, tipping points, or changes in complexity.

We formulated two main challenges of the microbiome dynamics analysis: 1) The complexity of microbial interaction and the temporal aspect of it and 2) Extracting the important microbial biomarkers that are associated with differentiation of stable microbiome composition (healthy) and the microbiome composition in alternative states (under external perturbations such as a disease or antibiotic exposure). The focus of this chapter is on the first sub-problem which will lead to tackling the second sub-program. In order to efficiently learn a low-rank representation of a time-evolving graph constructed from thousands of interactions of the microbiome, we leverage a

state-of-the-art model, the so-called Transformer. This model captures the temporal dependencies and changes in the graph structure by using an attention mechanism. We described the key parts of the Transformer and gave an extensive description of the idea behind the attention mechanism. Moreover, the Transformer updates node representations over time adapts to changing graph topologies, and propagates information through graph edges. The node representation is a common task in the machine learning task, however, in this work we are interested in learning the graph representation. For this reason, we integrate a special node, which we call a master node, that is connected to all other nodes in the graph. The low-rank representation of the master node is considered to be a low-rank representation of the whole time-snapshot graph. In order to make the model maintain the original dynamics of the microbiome, such as metastability, we propose to combine the Transformer with contrastive learning. The idea of contrastive learning is to encourage the model to bring similar data points closer in the learned representation space while pushing dissimilar data points apart. The common challenge of contrastive learning is defining positive and negative pairs, therefore we proposed an assumption for our problem based on the definition of metastability. Finally, we discussed the importance of positional encoding for our problem by bringing an example of a case where it is difficult for the model to distinguish different metastable states without positional encoding.

For our experiments, we continued working with the same datasets that we used in the previous chapter. However, we extended the synthetic data with one use case to show that if different metastable states have distinguishable graph features, the model does not require positional encoding. Therefore, we had three synthetic datasets with different configurations such as the number of nodes, the number of time steps, and the number of metastable states, and two real-world microbiome datasets. Through an extensive set of experiments on both synthetic and real-world datasets, we have demonstrated that our approach is capable of projecting a time-evolving graph into a low-dimensional space retaining the metastable properties of the system. We evaluated our results based on visual assessment and comparative analysis using an ARI metric. We compared the performance of our model with other dimensionality reduction methods and graph representation models. Moreover, we have illustrated the application of the proposed approach to microbiome data that enhances the analysis of metagenomic data in a way that takes into account a huge number of interactions among species.

There are several limitations of our model that should be mentioned here. The

first limitation is that we impose a heavy assumption on the dynamical properties of a time-evolving graph. The time-evolving graph has to display metastable behaviour as it is our main assumption in the contrastive loss: the current time-snapshot graph is more similar to the consecutive time-snapshot graph than to the time-snapshot graphs at any other time points. Moreover, we have not explored the importance of the history length parameter which can influence the training procedure since we use the accumulative gradient approach over to speed up the computation.

One of the future works is to investigate the fine-turning approach in the case when the size of data is small as it was in the case of the CholeraInf dataset. We noticed that the results for the CholeraInf changed from iteration to iteration. One solution for that can be to train a model on a large dataset, for example, the synthetic data with presumably two metastable states, and then fine-turn the model on the CholeraInf. Furthermore, throughout our experiments, we observed that the computational time required for other graph neural networks is considerably longer when compared to our proposed model, however, this requires a more vast set of experiments which can be done in future work. Another direction for future work can be the investigation of other dynamics such as coherent sets.

Chapter 4

Revealing high-impacting modules of microbiome via interpretability

An improved understanding of how the microbiome contributes to health and well-being can drive and accelerate the development of microbiome-based treatments. Yet, with the large variety of species, their complex interactions, and overall challenges of the microbiome dynamic analysis, it is rigorous for researchers to analyze the responses of the microbiome to different perturbations such as diseases or antibiotic exposures and their influence on human health. Up to this point, we have proposed two unsupervised methods which, first of all, take into account both multiple interactions of species of the microbiome and their temporal changes and second of all, put forth a strategy of simplifying the extraction of the microbiome dynamics. The underlying idea of our approaches to the analysis of the microbiome data is that instead of learning its dynamics in a high-dimensional space, we suggest projecting it onto a low-dimensional space in such a way that it is feasible to analyze the dynamics in this new space. To do so, the first method proposed in Chapter 2 utilizes the transfer operator theory and graph kernels to analyze temporal and structural changes in a time-evolving graph that represents the microbiome. The deep learning-based method introduced in Chapter 3 is based on the Transformer architecture and contrastive learning that also facilitates the extraction of temporal and structural properties of the time-evolving graph.

However, one of the most important questions, which has not yet been answered, is which species or interactions of species might be responsible for or affected by the changes that the biological interaction network undergoes from one state (healthy) to another state (diseased or antibiotic exposure). The presence of such valuable information can improve modern treatments of various diseases significantly. Furthermore, Section 1.1 presented the challenges that the microbiome analysis faces and it was mentioned that feature importance is an important component of the analysis. Finding microbial biomarkers that are associated with changes in a microbial composition is greatly connected to microbiome dynamics. Therefore, we aim to extend our methods proposed in previous chapters by introducing approaches for identifying graph structural features (or microbial biomarkers) that are associated with the transitions between metastable states. Although both our methods have their own advantages and limitations, which will be discussed in the next chapter, this chapter will focus on one of the key advantages of our proposed deep-learning method.

Conventional machine learning methods are claimed to be black boxes and thus, it is difficult to interpret the results of these methods. Moreover, as machine learning

models get more complicated and find their way into important areas like the field of medicine, it is essential to be able to interpret their outcomes to gain insights into how they are making predictions and to ensure their reliability and fairness. A large variety of methods have been proposed to explain and interpret the decision-making of machine learning-based methods. Although the key objective of the interpretability of machine learning is to gain insights into the internal workings of the model, understand the factors that contribute to its output, and be able to provide meaningful explanations for its behaviour, we will use interpretability methods as an approach to obtain the important biological components of the microbiome. This will improve our understanding of which species of the human microbiome are responsible for its transition from a stable state to an alternative state (under the influence of different perturbations e.g. diseases, and antibiotic exposure). For the approach proposed in Chapter 2, since most of the graph kernels use feature mapping that cannot be defined explicitly, it is impossible to go from the low-dimensional space back to the space of the time-evolving graph and extract structural features associated with metastable states. Therefore, we will use a simple averaging over time-snapshot graphs belonging to one metastable state. Since we have a finite set of metastable states and according to the definition of metastability 2.3.1, the structure of graphs within one metastable state should be similar. With this simple method, we will not be able to uncover more complex structural patterns such as walks. We leave it for future work.

This chapter will first discuss the main idea behind interpretability and some state-of-the-art methods. Then, we will present two approaches that we use to extract structural patterns of a time-evolving graph associated with metastable states. Finally, through a set of experiments on both synthetic and real-world datasets, we demonstrate the performance of the proposed approaches.

4.1 Model interpretation

With the massive success of machine learning (ML), there has been a need for interpreting these models, especially in the medical domain where decision-making significantly impacts human health and life. Gaining a better understanding of ML problem-solving strategies provides better communication so that we, as humans, can trust the model prediction. Moreover, trusting individual predictions is essential

to measure the robustness of the model before deployment.

The terms *interpretation* and *explanation* are sometimes used interchangeably. However, recent studies differentiate these terms by what kind of information they provide. The explanation is thought to generate outputs on the level of the system so that it provides more information as meaningful as it is possible for humans to comprehend. Interpretation in turn provides information on the level of the model, for example, which features contribute to a specific output. The interpretation output is not always understandable from the user's perspective (Hanif et al., 2021). In this thesis, we focus on interpretation methods rather than on explanation methods. To this end, an evaluation is done using accuracy metric or other metrics on the validation dataset. However, this raises a question of how we can trust this metric when the model is deployed. How can we be sure that the validation dataset contains all real-world cases and is generalized enough so that we can trust the decision-making of the model in real settings? If the interpretability explains that the model makes an individual prediction according to relevant variables, we can more reliably take actions according to the prediction. In general, interpretability is not needed in two situations (Saeed and Omlin, 2023): 1) if the results that are unacceptable do not lead to severe consequences and 2) if the problem has been studied in-depth.

Every year a large number of studies on AI interpretability are published. Moreover, various review studies are published covering a range of general and specific questions of XAI (Saeed and Omlin, 2023; Islam et al., 2021; Doshi-Velez and Kim, 2017; Hanif et al., 2021; Chaddad et al., 2023; Carrillo et al., 2021). The need for the XAI approaches can be viewed from five different perspectives: end-user, regulatory perspective, scientific, model development, and industrial perspective. All these consider the need for the XAI from different dimensions. We consider our work as a part of a scientific perspective since we aim at discovering novel concepts.

Even though the main reason for the successful development of the interpretability of machine learning models relates mainly to the trustworthiness and accuracy of the prediction, we utilize this approach to detect modules in the time-evolving graph. We make an assumption that relevant topological patterns of the time-evolving graph for the model are the modules of the time-evolving graph that are unique for each metastable state. Namely, when learning a low-dimensional representation the deep learning model should take into account the topological or attribute patterns of each metastable state.

Recently, different post-hoc explainable methods have been developed. The fol-

lowing subsection defines and introduces various post-hoc explanation methods. In this thesis, we will use the terms interpretability and explainability interchangeably, despite many articles that have attempted to differentiate these two terms.

4.1.1 What is an Interpretation?

The primary aim of the interpretation is to understand and explain the decisions, predictions, and behaviours of machine learning models. Since machine learning models become more complex and are used in critical applications, it is essential to be able to interpret their outcomes to gain insights into how they are making predictions and to ensure their reliability and fairness. All interpretation techniques can be divided into those that are inherently explainable methods (linear regression, decision tree-based models, generalized linear models), and those that require additional approaches to be interpretable. The latter can be further divided into model-agnostic methods, which allow the explanation method to be compatible with a variety of models, and example-based explanations which use specific data samples from the input data to explain the behaviour of the model in a model-agnostic way (Islam et al., 2021).

Furthermore, the model-agnostic interpretation approaches can be split into local and global methods. The local model-agnostic models describe the model behaviour in a target neighbourhood and measure if single predictions are made for the right reasons. For instance, in the binary classification problem, a local interpretability method should highlight discriminating features of the input between samples of two groups to show whether the reasoning of the models is right or wrong. Regarding our application, these discriminating features of the input are related to the perturbation. Moreover, if we do not know the discriminating features of metastable states, the explanation of the model can give insight into the unknown topological patterns in the time-evolving graph. The global surrogate models approximate the overall behaviour or decisions made by a more complex or black-box model. For example, deep neural networks or complex ensemble models are considered to be black-box models which are difficult to interpret due to a large number of parameters. Global model-agnostic methods are simplified and interpretable methods that approximate the behaviour of the aforementioned black-box models. It is important to note that using global surrogate models comes at the cost of potentially losing some accuracy or performance compared to the original black-box model. For instance, the behaviour

of a Random Forest can be approximated with classification and regression trees (Islam et al., 2021).

This thesis elaborates on the concepts above by adopting the local explanation methods for extracting important microbial features that are associated with the transition from a stable state to an alternative state. By employing local explanation methods, you aim to identify and highlight the crucial microbial features responsible for driving this transition. This could potentially provide valuable insights into the underlying mechanisms and factors influencing these microbial shifts. First, we will discuss different local and global interpretation methods for machine learning models.

Siliency map-based approaches

Saliency maps are visualization techniques to gain insights into the decision-making of machine learning methods. They were first proposed to generate an understandable visualization of deep convolutional network classification models.

The first gradient-based approach for saliency map visualization was introduced in (Simonyan et al., 2013). This method investigates how much a unit change in an input dimension induces in the output. Let f be a function that maps the input data $x \in \mathbb{R}^d$ to the output $y \in \mathbb{R}^C$, where C is the number of classes in the standard classification task. The salience map takes the gradient of class-specific logit with respect to the input i. (Springenberg et al., 2014) proposed a new method, *Guided backpropogation*, for image recognition using the deconvnet layer for activation feature visualization. They found that without max-pooling, the visualization of the discriminative features does not work very well using a deconvolutional network.

Another well-known method is *Class activation mapping (CAM)* proposed by (Zhou et al., 2016). They argue that global average pooling, which was thought to act as a regularize, with small turning, makes the network retain its localization ability until the final layer. *Gradient-weighted Class Activation Mapping or Grad-CAM* (Selvaraju et al., 2017) is another method for saliency map visualisation. The idea is to use the gradient of any target class flowing into the final convolutional layer to produce a coarse localization map that highlights the important regions in the image for predicting the concept. Formally, the final activation map can be expressed as follows:

$$L^c = ReLU\Big(\sum_k \alpha_k^c A^k\Big) \tag{4.1}$$

with a neuron importance weight:

$$\alpha_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k}, \tag{4.2}$$

where the index c is a class, the index k denotes the index of the feature map for target class c, and A^k is a feature map activations of a convolutional layer.

All these gradient-based methods suffer from a saturation problem (Shrikumar et al., 2017). Using the ReLU activation function results in that the gradient coming into a ReLU during the backward pass becomes zero if the input to the ReLU during the forward pass is negative. Due to zeroing out of negative gradients, the gradient-based will most likely identify neurons with saturated activation as non-important.

Finally, it is worth mentioning that although all these approaches have been proposed mainly for the interpretation of a model consisting of convolutional layers on image data, there have been done several studies that apply these methods to other deep learning architectures such as graph convolutional layers (Arslan et al., 2018; Pope et al., 2019).

Layer-wise Relevance Propagation

Layer-wise relevance propagation (LRP) (Bach et al., 2015) is the family of explanation methods that leverages the layered structure of the network. The initial variation of this method can also potentially be attributed to the saliency map-based methods as it also produces a visualization of image features that contributes to the decision-making of the model. It explains the prediction of a neural network classifier by backpropagating the neuron activation on the output layer to the previous layers until the input layer is reached. Such redistribution results in the attribution of the predictions to the different input features. This propagation procedure is subject to a conservation law, where what has been received by a neuron must be redistributed to the lower layers in equal amounts:

$$f(x) = \cdots = \sum_{d \in l+1} R_d^{(l+1)} = \sum_{d \in l} R_d^{(l)} = \cdots = \sum_d R_d^{(1)}, \tag{4.3}$$

where f is the classifier, the index l indicates a layer, $R_d^{(l)}$ is a relevance score on the layer l. Formally, LRP can be defined in the following way. Given j and k neurons at two consecutive layers of the neural network, the relevance score at the layer l is

computed in the following way:

$$R_j = \sum_K \frac{z_{jk}}{\sum_j z_{jk}} R_k, \tag{4.4}$$

with z_{jk} being the quantity that models the extent to which neuron j has contributed to make neuron k relevant.

There are several different LRP rules for feedforward networks, which find their mathematical foundation in Deep Taylor Decomposition (DTD) (Montavon et al., 2017).

Basic rule (LRP-0). (Bach et al., 2015). This rule involves redistributing contributions proportionally from each input to the activation of the neuron

$$R_j = \sum_k \frac{a_j w_{jk}}{\sum_{0,j} a_j w_{jk}} R_k. \tag{4.5}$$

It can be shown that for ReLU-activated neural networks (defined in Eq. 3.5), the LRP-0 rule produces an explanation equivalent to Gradient × Input discussed in Section 4.1.1. However, as it was said the gradient-based methods suffer from shattered gradients and adversarial examples (Montavon et al., 2019).

Epsilon rule (LRP-ϵ). (Bach et al., 2015). This rule was proposed to improve the LRP-0 rule which consists of adding a small positive term ϵ in the dominator:

$$R_j = \sum_k \frac{a_j w_{jk}}{\epsilon + \sum_{0,j} a_j w_{jk}} R_k. \tag{4.6}$$

The idea of the term ϵ is to absorb some relevance, when the contributions to the activation of neuron k are weak or contradictory. When ϵ is larger, only the most silent explanation relevance scores survive the absorption.

Gamma rule (LRP-γ). (Bach et al., 2015). This is another enhancement that was introduced to prioritize the influence of positive contributions over negative contributions:

$$R_j = \sum_k \frac{a_j \dot{(} w_{jk} + \gamma w_{jk}^+)}{\sum_{0,j} a_j \dot{(} w_{jk} + \gamma w_{jk}^+)} R_k. \tag{4.7}$$

The parameter γ controls how much positive contributions are prioritized. With increasing γ, the negative contributions disappear. This helps to develop more stable explanations.

Alpha-beta rule (LRP-$\alpha\beta$). (Bach et al., 2015). Similar to the LRP-γ rule, it allows controlling the importance of positive and negative values.

$$R_j = \sum_k \Big(\alpha \frac{(a_j w j k))^+}{\sum_{0,j}(a_j w_{jk})} - \beta \frac{(a_j w_{jk})^-}{\sum_{0,j}(a_j w_{jk})^-} \Big) R_k. \tag{4.8}$$

Here, the non-negative α parameter permits a weighting of relevance distribution towards activations and inhibitions and the commonly used value of α is one. The parameter β is given implicitly such that $\alpha+\beta = 1$. Moreover, empirically, it has been shown that the LRP-$\alpha\beta$ rule demonstrates robustness against gradient shattering and produces visually pleasing attribution maps (Kohlbrenner et al., 2020).

Initially, this redistribution-based approach with different rules was primarily suggested for networks based on feedforward or convolutional architectures. Recently, the LRP paradigm has been extensively applied to other neural network architectures such as graph neural networks (Xiong et al., 2023; Xiong et al., 2022; Schnake et al., 2021), Transformers (Ali et al., 2022; Chefer et al., 2021b), similarity models (Eberle et al., 2020), recurrent neural networks (Samek et al., 2022) and LSTM (Arras et al., 2019), which also enables explaining different data structures like graphs, text or time-series data. Therefore, since the focus of this thesis is mainly on graph-structured data, we think it is worth discussing the LRP explanation method for GNN. As a reminder, GNNs are typically constructed by stacking several interaction blocks. Each block computes a node representation $H_b \in \mathbb{R}^{n \times d_b}$, where b is an index of a block, n is the number of nodes in the input graph and d_b is the number of dimensions of a node representation vector. The whole input-output relation implemented by GNN can be defined as:

$$f(\Lambda; H_0) = g(H_B(\Lambda, H_{B-1}(\Lambda, \ldots, H_1(\Lambda, H_0)))), \tag{4.9}$$

where Λ is the input graph given as an adjacency matrix of size $n \times n$, g is a readout function (Gilmer et al., 2017), and H_0 is an initial state. Then, the relevance

propagation rule for a graph convolutional network can be expressed as:

$$R^a_{JKL\ldots} = \sum_b \frac{\lambda_{JK} h^a_j w^*_{ab}}{\sum_{J,a} \lambda_{JK} h^a_j w^*_{ab}} R^b_{KL\ldots} \tag{4.10}$$

with w_{ab} denoting the element of the matrix W that links neuron a to neuron b and $w^* = w + \gamma w^+$. However, similar explanation rules can be defined for other GNNs. The big advantage of this method is that it can identify not only relevant node or edge attributes, but it can also extract more complex structural features like walks.

Local Model-Agnostic Methods

This category of explanation methods does not make any assumptions about the internal structure of the model and can be applied to any machine learning method. The first method from this category is a local interpretable model-agnostic explanation (LIME) (Ribeiro et al., 2016) is a so-called local surrogate model that approximates the prediction of the underlying black-box model. The intuitive idea of LIME is the following. Given only the black-box model without the training data, LIME tests what happens in the prediction of the model when you give variations of data. It generates new data with perturbed samples and the corresponding predictions of the black-box model. It then trains an interpretable model on the new dataset, which is weighted by the proximity of the sampled instance to the instance of interest. The interpretable model should be a good approximation of the predictions of the black-box model locally, but it does not have to approximate well globally. Formally, LIME can be defined as follows:

$$R(x) = \underset{g \in G}{\operatorname{argmin}}\, L(f, g, \pi_x) + \Omega(g), \tag{4.11}$$

where x is a data instance for which we want to obtain an interpretation, g is an interpretable model, f is a black-box model, $\Omega(g)$ is the model complexity which we want to keep as low as possible, and G is a family of possible explanations. Finally, π_x is a proximity measure that says how large the neighbourhood of the instance x is that we consider for the interpretation.

Another method is SHapley Additive exPLanations (SHAP) (Lundberg and Lee, 2017). This method uses Sharpley values to explain the output of the model and is attributed to the class of additive feature attribution methods. This method can

be considered to be both global and local. The goal of the method is to explain the prediction of an instance x by computing the contribution of each feature to the prediction. Let g be the explanation model, $z' \in \{0,1\}^M$ is the "simplified features", then the explanation can be expressed as:

$$g(z') = \psi_0 + \sum_{j=1}^{M} \psi_j z'_j, \tag{4.12}$$

where ψ is the feature attribution for a feature j or the Shapley value.

Global Model-Agnostic Methods

Global explanations are designed to explain the complete and average behaviour of a given model to provide a bird-eye-view of the model. Instead of producing an explanation of one specific model prediction or one instance of input data, global explanation methods propose an approach how to generate an explanation, for example, for all model predictions. One of the first methods from this category was introduced in (Craven and Shavlik, 1995). It is a surrogate model which meant that it proposes another interpretable method to explain the original model. Particularly, they use a decision tree to approximate a trained network.

Testing with Concept Activation Vectors (TCAV) (Kim et al., 2018) is another global method that provides an explanation of an internal state of a neural network model in terms of human-friendly. It uses derivatives to estimate to which extent a user-defined idea is important to the model prediction. The first step of TCAV is to define a set of instances with "concepts" of interests that TCAV aims at learning. The objective of the second step is to find a concept activation vector. It is done by learning a linear classifier to distinguish between the activation produced by a concept's examples and examples in any other layer. Then a concept activation vector is defined as the normal to a hyperplane separating examples without a concept and examples with a concept in the model's activation. Finally, directional derivatives are calculated to measure the sensitivity of the concept.

(Yang et al., 2018) introduced a Recursive Partitioning method for the global interpretation of black-box models. They suggest a compact binary tree to explicitly represent the most important decision rules that are implicitly contained in the black-box machine learning models. This tree is learned from the contribution matrix that consists of contributions of each input variable to the predicted score

for each prediction made by the model. To generate a compact binary tree, they propose a unified process that recursively partitions the input space by maximizing the difference in the average contribution of the split variable between the divided spaces.

4.2 New Feature Extraction Methods for Microbiome Data

In this section, we will discuss interpretability methods that we use to obtain the associated graph features for metastable states. The first method will be the LRP-based method from (Chefer et al., 2021b). We follow this method with some adjustments that will be discussed further in the chapter. Another method is the attention-based method. It has been proved that the attention weights can be considered a faithful explanation of the model prediction.

4.2.1 Problem formulation

Given a time-evolving graph $\mathcal{G}$ as a sequence of T graphs $\mathcal{G} = (G_0, \ldots, G_{T-1})$ at the consecutive time points $\{0, \ldots, T-1\}$ for some $T \in \mathbb{N}$. We assume that the time-evolving graph exhibit metastable behaviour and each metastable state is characterized by a certain structural pattern of the graph such as node clusters, edge clusters, or walks. Moreover, let f be a function that learns a low-dimensional representation of the time-evolving graph $\mathcal{G}$ such that $f(\mathcal{G}) = \mathbf{g}$, where $\mathbf{g} = \{g_1, \ldots, g_T\}$ with $g_i \in \mathbb{R}^d$ and each $g_i \in \mathcal{G}_s$ with s being the number of metastable states. We assume that the function f is represented by one of our methods, graphKKE from Chapter 2 or deep learning-based method from Chapter 3. We can formulate the problem in the following way: *Given a low dimensional space $\mathbb{S}$, we aim at finding a set of graph features $\mathcal{X} \in \mathbb{G}$ (node clusters, edge clusters, or walks) which are associated with transitioning from one metastable state to another*

This chapter will be mainly related to the deep learning-based method, however, we will also propose a simple approach to extracting relevant graph features for graphKKE.

4.2.2 LRP-based approach for biomarkers extraction

There are not many contributions that explore the interpretability of Transformer-based architectures. In order to explain the output, pave the way for a better explanation of our model, and obtain graph features associated with metastable states, we follow the idea of explaining Transformer-based models introduced in (Chefer et al., 2021b). They employ a class-specific visualization of self-attention models by computing the final relevance scores by multiplying the relevance score of each Transformer layer with the gradient of the corresponding attention matrix summed up across the "head" dimension. Moreover, the authors address the lack of conservation property in the attention mechanism because of matrix multiplication and the numerical issues of the skip connections by applying a normalization to the computed relevance score.

In order to address the lack of conservation in the attention mechanism and skip connections, (Chefer et al., 2021b) apply normalisation to the relevance of two feature map tensors, skip connection and attention matrix as:

$$\bar{R}_j^{u^{(n)}} = R_j^{u^{(n)}} \frac{\left|\sum_j R_j^{u^{(n)}}\right|}{\left|\sum_j R_j^{u^{(n)}}\right| + \left|\sum_k R_k^{v^{(n)}}\right|} \cdot \frac{\sum_i R_i^{(n-1)}}{\sum_j R_j^{u^{(n)}}}$$

and

$$\bar{R}_k^{v^{(n)}} = R_k^{v^{(n)}} \frac{\left|\sum_k R_k^{v^{(n)}}\right|}{\left|\sum_k R_k^{v^{(n)}}\right| + \left|\sum_j R_j^{u^{(n)}}\right|} \cdot \frac{\sum_i R_i^{(n-1)}}{\sum_j R_k^{v^{(n)}}}$$

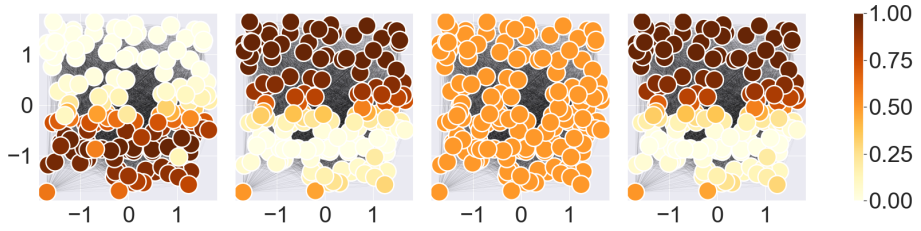
where $R_j^{u^{(n)}} = \mathcal{D}(u, v, R^{(n-1)})$ and $R_k^{v^{(n)}}) = \mathcal{D}(v, u, R^{(n-1)})$ are the generic deep Taylor decompositions on two tensors u and v. The index j corresponds to elements in the input of layer $L^{(n)}$ with $n = \overline{1, N}$ as the layer index in the model.

Let $\mathbf{A}^{(b)}$ be the attention matrix of a Transformer block b. We first compute the weighted attention relevance as follows:

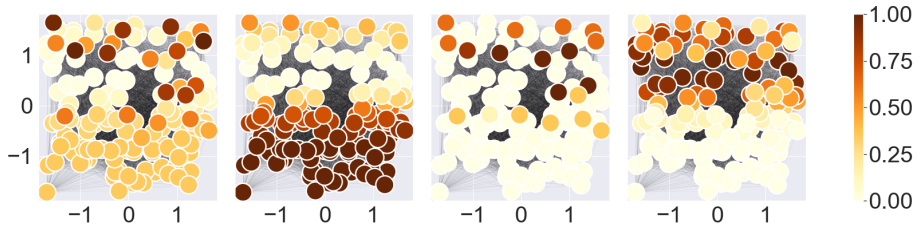
$$\bar{A}^{(b)} = I + \mathbb{E}_b(\bigtriangledown A^{(b)} \odot R^{(n_b)}). \tag{4.13}$$

Then, the final relevance score of the method is defined as:

$$R = \bar{A}^{(1)} \dots \bar{A}^{(B)}, \tag{4.14}$$



(a) State 1



(b) State 2

Figure 4.1: Fully-connected graphs for the *npos_2WellGraph* dataset with nodes colour-coded based on attention weights for each of four heads in the Transformer. Attention weights are extracted from the model while learning the low-dimensional representation of each time-snapshot graph and summed across two states. The locations of two states are obtained by clustering points of the final low-dimensional representation via k-means.

where $\odot$ is the element-wise product, B is the number of heads in the Transformer, $\mathbb{E}_h$ is the mean across the "heads" dimension. The identity matrix I is added to account for the skip connections in the Transformer block. The attention weight matrix $A^{(b)}$ is normalized with *softmax* so that $\sum_{ij} A_i j^{(b)} = 1$.

Unlike original LRP and (Chefer et al., 2021b), where the decomposition starts from the classifier output corresponding to the target class, we have a similarity model that measures how similar resulting low-dimensional representations of the time-snapshot graphs G_t and G_{t+1} are. For this reason, we start the redistribution from the layer, where we have computed the low-dimensional representation $\hat{g}_t$, that is:

$$f(x) = \hat{g}_t,$$

then we redistribute using Eq.4.4 until the input layer is reached, and the final relevance $R^{(1)}$ is computed.

After having obtained a low-dimensional representation of the time-evolving graph, we cluster each point of it with k-means clustering. We then compute a relevance score for each time-snapshot graph in the test set. In order to obtain dis-

Table 4.1: Statistics of each dataset used in the Chapter 4.

Name	#Nodes	#Edges	#Time steps	#States
pos_5WellGraph	100	4851	500	5
npos_2WellGraph	150	10821	10000	2
CholeraInf	96	106	34	2

criminating features of the whole state, we sum up relevance scores of time-snapshot graphs of each state:

$$R_s = \frac{1}{|T_s|} \sum_{i \in T_s} R_i^{(1)},$$

where s is an index of the state, T_s is a set of indices of time-snapshot graphs from the state s, $R_i^{(1)}$ is a matrix of final relevance scores of the i-th time-snapshot graph.

4.2.3 Attention as interpretability

One of the critical components that make the Transformer architecture so successful is the attention mechanism. The introduction of attention mechanisms has greatly improved the performance of neural network models in handling sequences of variable lengths, allowing the model to focus on relevant parts of the input during the computation. Due to the attention mechanism, the Transformer architecture has revolutionized the field of NLP and has been widely adopted due to its ability to capture long-range dependencies in sequences more effectively than traditional recurrent neural networks. Moreover, it is worth noting that the success of attention mechanisms is not limited to NLP alone. They have also been applied in other domains such as computer vision, where they have been used to improve tasks like image captioning, visual question answering, and object detection.

In Chapter 3 we have given a vast overview of how the attention mechanism emerged and what makes the attention mechanism so effective. Knowing the idea behind attention and its revolutionized impact on the machine learning domain, it becomes imperative to explore whether the attention matrix can enhance the interpretability of deep learning models or if it can be used to extract important features from the input data and conduct the so-called feature importance analysis. In the case of the microbiome and, in general, in the medical domain, which involves high-risk decisions that impact human health and life, it is essential to know if and

when to trust model predictions.

There have been published a lot of controversial papers on whether or not the attention weights can be considered as an explanation. The first work arguing that the attention mechanism cannot be utilized as a way to explain the decision-making of the model is (Jain and Wallace, 2019). They give insight into whether some relationship between the attention weights and model outputs exists. Throughout their experiments, they find that in fact, there is no association between attention weights and model outputs. The first experiment analyzes the correlation between gradient-based feature importance scores and learned attention weights. Based on this experiment, they have not found any strong agreement of attention weights with standards feature importance scores. The second experiment includes randomly permuting attention weights and recording corresponding changes in model outputs. The results of the second experiment indicate that learned attention weights are uncorrelated with gradient-based measures of feature importance, and it is possible to define any other attention distributions which will yield almost the same model prediction. Another work (Serrano and Smith, 2019), which supports the previous work, bases its experiments on removing intermediate representations to explore whether attention weights can be used to explain the relative importance of the inputs to the attention layer itself. As a result, they find that attention does not necessarily corresponds to importance.

The following set of publications (Clark et al., 2019; Wiegreffe and Pinter, 2019; Hao et al., 2021; Chefer et al., 2021a; Chefer et al., 2021b) is challenging the results discussed in the previous paragraph by arguing that in fact, attention can be considered as an explanation of the decision-making of the model. For instance, (Wiegreffe and Pinter, 2019) claim that the assumptions made in (Serrano and Smith, 2019) depend on a definition of explanation, and testing if the attention can be considered as an explanation requires taking into account all elements of the model. Their set of experiments shows that even in the case of adversarial attention distribution they do not perform as well as original attention weights which implies that we cannot disprove the usefulness of the attention mechanism for interpretability of the model. Other above-mentioned works focus on developing interpretability methods that use the attention mechanism. On the contrary, it has been shown that the gradient in a Transformer reflects the function only locally and thus, fails to identify the contribution of input to the prediction (Ali et al., 2022). In turn, they demonstrate that attention and LayerNorm are components of the Transformer that

lead to unreliable explanations.

Some works try to answer the question of why attention might not be suitable for interoperability. (Bai et al., 2021) claims that the problem of attention mechanism as a way of interpretability is the combinatorial shortcuts. The main assumption for the effectiveness of attention-based interpretability is that the attention mechanism carries only information from the highlighted parts of V. However, since the attention mechanism contains the products of the masks and V, the masks can carry extra information. (Liu et al., 2022) explores the problem of the polarity of feature impact: features with higher attention weights may not necessarily contribute to the model prediction. The main idea of this paper evolves around the faithfulness of an explanation that has to fulfil the following: 1) importance correlation and 2) polarity consistency. They propose a *faithfulness violation test* which assesses if features with the largest explanation weights demonstrate consistent polarity. Further, we will discuss how we use the attention weights to extract important microbial biomarkers associated with each metastable state.

The research on this topic has not reached the level of maturity yet and although some works have shown that attention may not be a good way of interpretability of the model, we will nevertheless utilize it as a method of extracting important graph features that are associated with the transitions of the time-evolving graph from one metastable state to another. Moreover, the interpretability via attention will be compared to the results obtained with the LRP-based method.

We will follow the following procedure to obtain an attention-based explanation. Similar to the experiments with the LRP-based method, we first train the Transformer-based model presented in Chapter 3 on a time-evolving graph. We define a test set of time-snapshot graphs $\{G_t, \ldots, G_{t+i}\}$, where $i \in \mathbb{N}$ such that $t + i <= T$ and so that the test set contains time-snapshots graphs from all metastable states. Moreover, each point of the resulting low-dimensional representation of the test set is then assigned a label based on the k-means clustering. Finally, for each t, we extract a corresponding attention matrix H_t from the model. In order to obtain graph features associated with each metastable state, we compute the average attention weight matrix for each metastable state as follows:

$$\hat{H}_s = \frac{1}{T_s} \sum_{i=1}^{T_s} H_i,$$

where s is an index of the current metastable state, $H_i \in \mathbb{R}^{(n+1)\times(n+1)}$ is an attention

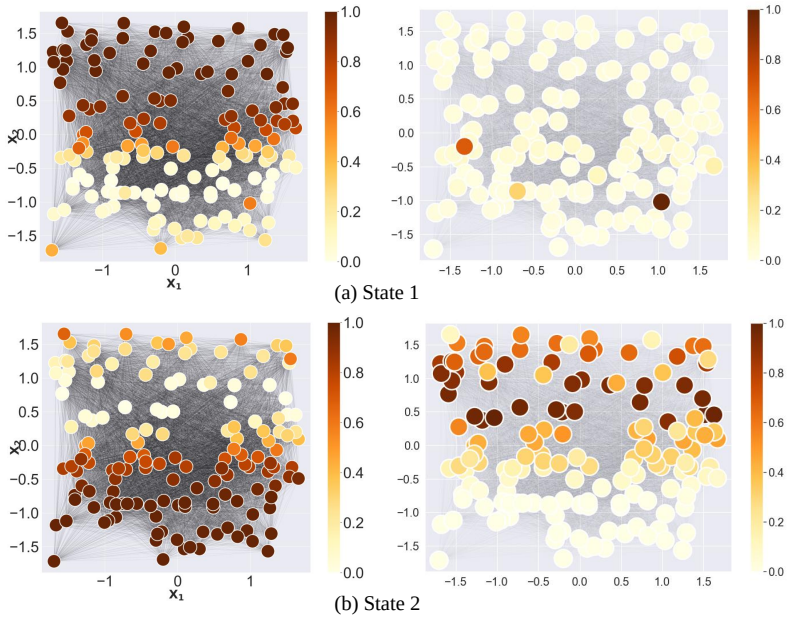


Figure 4.2: Fully-connected graphs for the *npos_2WellGraph* dataset with nodes colour-coded based on **(Right)** the relevance scores of the LRP-based method that are summed across 2 states and **(Left)** on the averaging method 4.15 for graphKKE. The locations of 2 states are obtained by clustering points of the low-dimensional representation of the time-evolving graph via k-means.

matrix extracted from the model with the input of the time-snapshot G_i and T_s is the number of time-snapshot graphs in the state s. Since $\hat{H}_s \in \mathbb{R}^{(n+1)\times(n+1)}$, where the $(n+1)th$ node is the master node which we use as a graph embedding of the entire time-snapshot graph, we assume that this node encapsulates an explanation of the whole time-snapshot graph. The $(n+1)$th entry of the matrix $\hat{H}_s$ is employed to colour-code nodes in a fully-connected graph G_0.

4.3 Experiments and Results

In this section, we will present how well the interpretability methods discussed in Section 4.2 can extract relevant graph features associated with metastable states.

4.3.1 Experimental setup

First, we will apply both graphKKE from Chapter 2 and a deep-learning-based model from Chapter 3 in order to obtain a low-dimensional representation of the

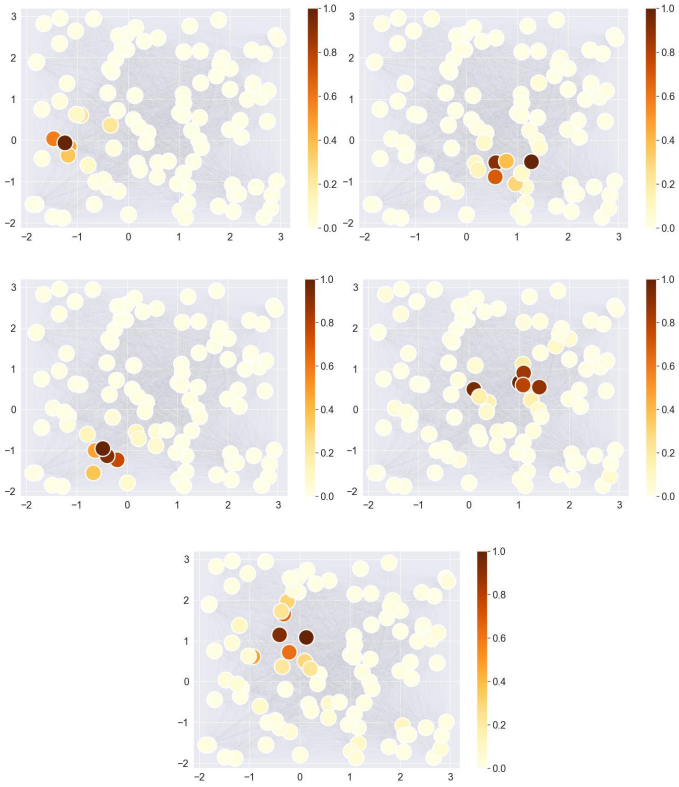


Figure 4.3: Fully-connected graphs for the *pos_5WellGraph* dataset with nodes color-coded based on the relevance scores of the LRP-based method that are summed across 5 states. The locations of five states are obtained by clustering points of the resulting low-dimensional representation via k-means.

time-evolving graph. For experiments in this chapter, we will use synthetic datasets from previous Chapters and one real-world dataset for both methods. For synthetic datasets, we will generate pos_5WellGraph and npos_2WellGraph, and for a real-world dataset, CholeraInf is utilized. The dataset statistics can be found in Table 4.1. After the low-dimensional representation for a corresponding time-evolving graph is obtained, k-means clustering is applied in order to identify the location of metastable states in the time-evolving graph.

In order to understand if graphKKE presented in Chapter 2 can also associate each metastable state with some graph features, we will propose a straightforward method for extracting these graph features. However, it is worth noting that this method will not be able to capture complex graph features such as walks. With this strategy, we also aim to compare the capabilities of the two methods presented in this thesis.

Applying k-means to the eigenfunctions associated with the five dominant eigenvalues results in the five clusters. Since each state of the time-evolving graph is characterized by some common pattern in the topological structure, we average the adjacency matrices of each state. Thus, if we have a time-evolving graph with s states $\mathcal{G} = \mathcal{G}_0 \cup \cdots \cup \mathcal{G}_{s-1}$ and $\{\mathcal{A}_0, \ldots, \mathcal{A}_{s-1}\}$ is a set of corresponding subsets of adjacency matrices, then

$$\mathcal{A}_i^{avg} = \frac{1}{|\mathcal{A}_i|} \sum_{j=0}^{|\mathcal{A}_i|-1} A_i^j, \tag{4.15}$$

where $A_i^j \in \mathcal{A}_i, i = 0, \ldots, s-1$. Each average adjacency matrix $\mathcal{A}_i^{avg}$ is associated with the average graph $\mathcal{G}_i^{avg}$.

It is worth noting that the method described above does not produce the graph features that are associated with metastable states or graph features that are important for the model to learn a low-dimensional representation. It outputs an average degree matrix and in the case of more complex relevant graph structural features, which is often the case for microbiome, the result of this method will not be reliable. On the contrary, the interpretation method proposed in Section 4.2 is able to produce graph features that are relevant to the model's decision-making. We assume that as they are relevant for the model, these graph features are the ones that differentiate metastable states.

4.3.2 Results: Synthetic data

In this part, we assess the interpretation results for the synthetic dataset. We begin with the results of the interpretation for the npos_2WellGraph dataset. *The average attention matrix* of each head obtained while learning a low-dimensional representation with the deep learning method proposed in Chapter 3 is given in Figure 4.1. As is evident from the visualization, the model focuses on both the upper and lower parts of the time-snapshot graph. These areas are also discriminating for us as humans to distinguish structural patterns of time-snapshot graphs in both states. It is interesting that the attention matrix of one state contains the information of another state, which is different in the case of LRP.

In the case of *the LRP-based approach* for npos_2WellGraph, the results are presented in Figure 4.2(Right). The interpretation when predicting state two (Figure 4.2(Right)(a)) highlights the upper part of the time-snapshot graph, which is a ground-truth graph feature associated with state 2 that was defined when generating

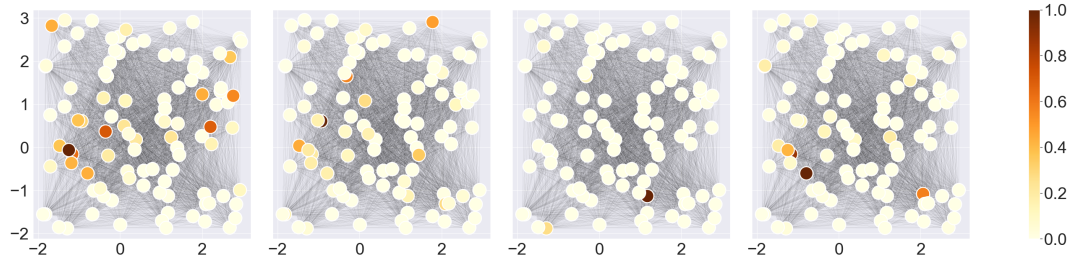
the dataset. However, the interpretation of state 1 (Figure 4.2(Right)(b)) shows only four importance nodes in the lower part of the time-snapshot graph.

Figure 4.2 (Left) demonstrates node clusters associated with metastable states when learning the low-dimensional representation with *graphKKE*. We can see the presence of the two node clusters: one in the upper part of the graph and another located in the lower part of the graph. For state two (Figure 4.2 (Left)(b)), the majority of nodes in brown colour are located in the lower part, while all nodes in brown colour for the experiment with the LRP-based method are located in the upper part. The difference is that in the case of the LRP-based model, we extract graph features that have the highest degree of association with metastable states. For the averaging method of graphKKE, however, we compute an average graph across each state that is defined by removing edges between certain nodes. Therefore, in this case, the cluster of nodes in the light colour indicates the graph feature that is associated with the corresponding state.

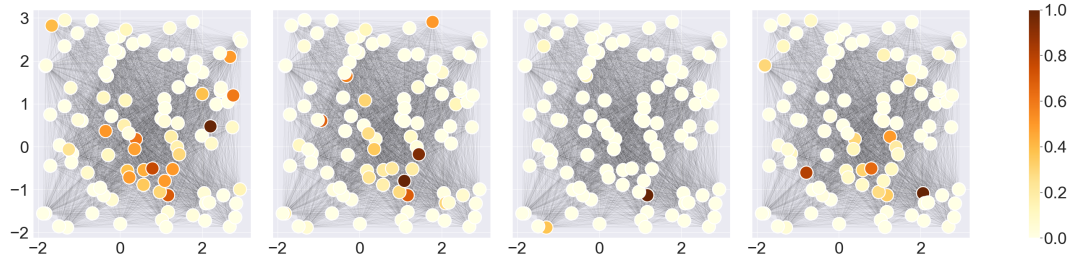
Let us examine the result of the last synthetic dataset which is pos_5WellGraph. As before, we discuss first the result of visualizing attention matrices, the result of the LRP-based method, and the result of the averaging method in Eq. 4.15. Figure 4.4 shows attention matrices of four heads for each of five states. As it was with npos_2WellGraph, the visualization of attention weights shows the presence of five different clusters of nodes, and some heads have information about the locations of discriminating graph features of other states as well. Unlike the result for npos_2WellGraph, the interpretation via the LRP-based method for pos_5WellGraph, shown in Figure 4.3, highlights five node clusters (brown) that are associated with each metastable state.

Figure 4.5 illustrates the result of the averaging method of graphKKE. We can also observe five different node clusters (light). Yet, these node clusters represent nodes with the lowest number of neighbours and not the importance of each of these node clusters for the model to learn a low-dimensional representation.

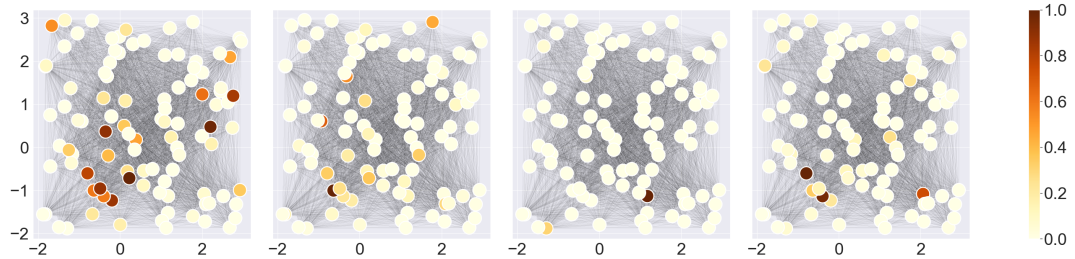
There is necessary to mention that we have modelled synthetic datasets in such a way that we know the location of relevant graph features associated with metastable states. In the real-world setting, we do not have the ground-truth relevant graph features and consequently, visualizing the interpretation results becomes challenging. These situations require further investigation in future studies. Additionally, it is noteworthy that the approach discussed in this chapter can offer valuable insights into novel biological aspects when applied to real-world microbiome data.



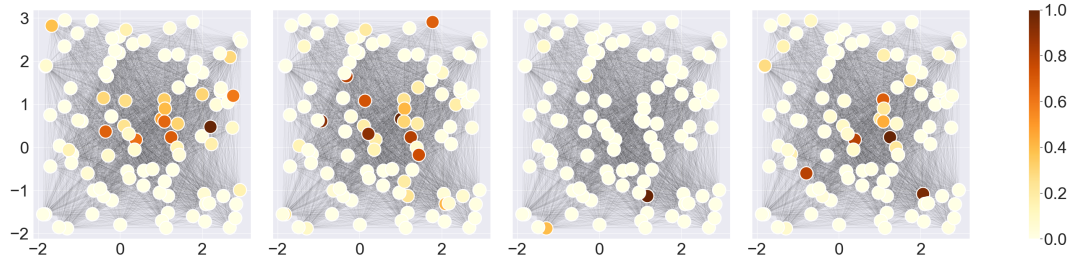
(a) State 1



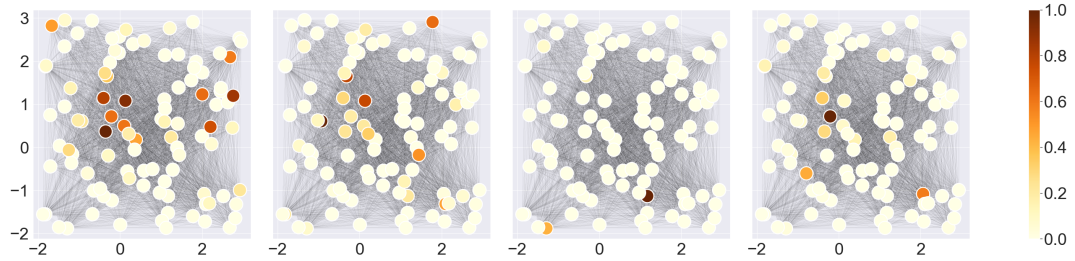
(b) State 2



(c) State 3



(d) State 4



(e) State 5

Figure 4.4: Fully-connected graphs for the *pos_5WellGraph* dataset with nodes colour-coded based on attention weights for each of four heads in the Transformer. Attention weights are extracted from the model while learning the low-dimensional representation of each time-snapshot graph and summed across states. The locations of five states are obtained by clustering points of the resulting low-dimensional representation of the time-evolving graph via k-means.

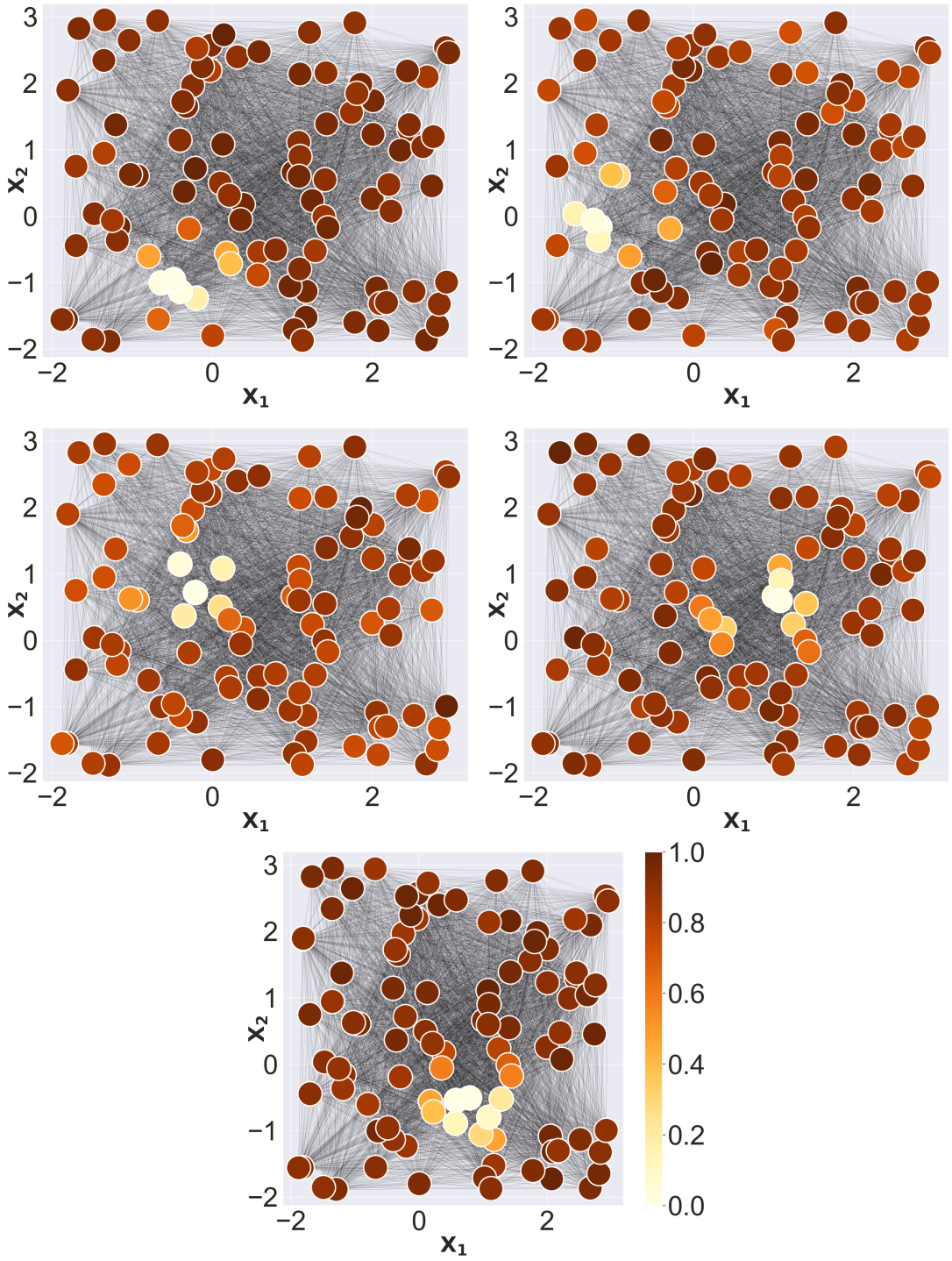
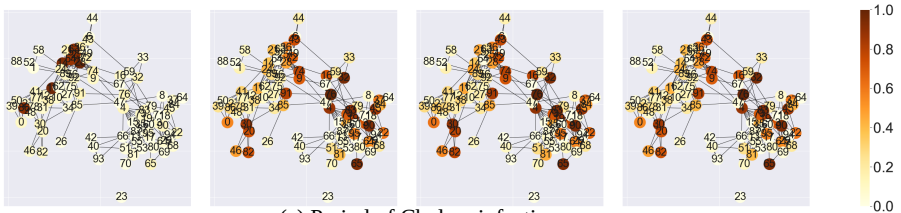


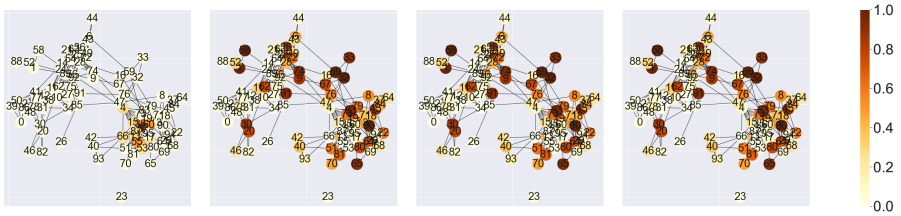
Figure 4.5: Fully-connected graphs for the *pos_5WellGraph* dataset with nodes colour-coded based on the averaging method in Eq. 4.15 for graphKKE. The locations of five states are obtained by clustering points of the resulting low-dimensional representation of the time-evolving graph via k-means.

Table 4.2: The results of the LRP-based approach obtained in Chapter 4. The list of microbial species that are associated with a period of cholera infection and a period of recovery in the *CholeraInf* dataset.

Node index	**Microbial species**
	Period of cholera infection
19	Proteobacteria-Acetobacteriaceae
20	Firmicutes-Veillonellaceae
22	Proteobacteria-LS4-241
30	Proteobacteria-SM2D12
32	Firmicutes-Bacillus
46	Proteobacteria-Methylococcaceae
64	Proteobacteria-Syntrophobacteraceae
76	Firmicutes-Thermoanaerobacteraceae
82	NPL-UPA2-unclassified
83	Thermodesulfobacteria-Thermodesulfobacteriaceae
90	Proteobacteria-Methylobacteriaceae
	Period of recovery
51	Verrucomicrobia-Opitutaceae
66	Firmicutes-Family_XIV_Incertae_Sedis
70	Proteobacteria-Methylocystaceae
79	Proteobacteria-LWSR-14
80	Proteobacteria-Sorangiineae
81	Spirochaetes-Leptospiraceae
95	Bacteroidetes-SB-1



(a) Period of Cholera infection



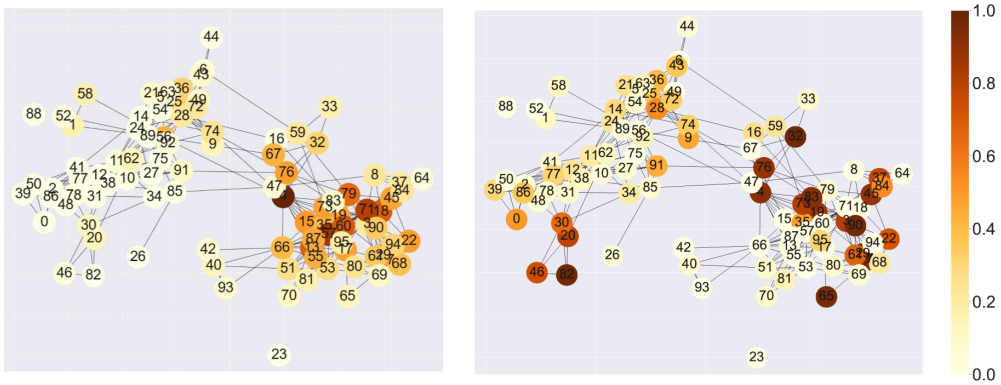
(b) Period of recovery

Figure 4.6: Co-occurrence interaction graphs of *CholeraInf* dataset with nodes colour-coded based on attention weights for each of four heads in the Transformer. Attention weights are extracted from the model while learning the low-dimensional representation of graph-snapshot graphs and summed across each state. The locations of states are obtained by clustering points of the resulting low-dimensional representation of the time-evolving graph via k-means. The dark brown colour indicates nodes with the highest attention weights.

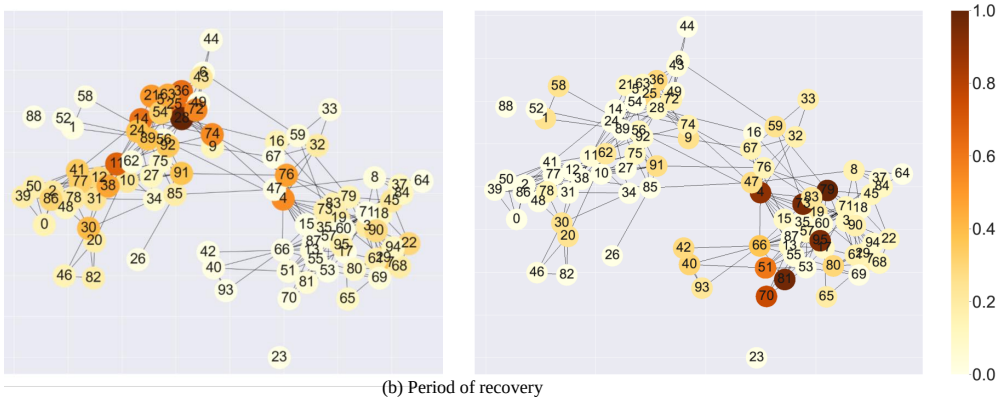
4.3.3 Results: Microbiome data

We have assessed the interpretation of the approach proposed in this Chapter on the synthetic dataset, and now we focus on obtaining the interpretation of the model's decision for the real-world dataset, namely, *CholeraInf*, which has been described in Section 3. Unlike the synthetic dataset, we do not know the ground-truth discriminating features for this data. Moreover, in order to visualize the interpretation, we use a correlation matrix that has been computed based on the OTU table (see more details about how this data has been pre-processed in Section 2.5.1). This dataset contains two metastable states: a period of cholera infection and a period of recovery. The ground-truth locations of metastable states are known from both original datasets and from the results obtained with the two methods proposed in this thesis. We begin the discussion of results with attention weights.

Figure 4.6 shows the interpretation assessment through the attention matrix. We can see that the attention weight of the first head of the Transformer differs from the attention weights of other heads. The attention weights of the second, third, and fourth heads are identical. It is evident that there are some nodes that are



(a) Period of Cholera infection



(b) Period of recovery

Figure 4.7: Co-occurrence interaction graphs of *CholeraInf* dataset with nodes colour-coded based on **(Right)** relevance scores of the LRP interpretation which are summed across each state and **(Left)** the averaging method for graphKKE. The locations of states are obtained by clustering points of the resulting low-dimensional representation of the time-evolving graph via k-means. The dark brown colour indicates nodes with the highest importance for the decision-making of the model.

different in their importance between Figure 4.6(a) and Figure 4.6(b). For example, nodes $\{1, 33, 58, 76\}$ have different importance when learning the low dimensional representation of time-snapshot graphs from different metastable states.

Figure 4.7(Left) depicts the result of the averaging method for graphKKE, which simply shows an average graph across metastable states. The results of the LRP-based method for both metastable states are presented in Figure 4.7(Right), where we have obtained two clear clusters of nodes. It can be seen that there are different nodes for both metastable states that have the highest importance for the model. For instance, nodes $\{32, 46, 65, 82, 90\}$ have the highest importance for a metastable state of cholera infection, while for the period of recovery, nodes $\{51, 70, 81, 79, 95\}$ are of the highest importance. We consider these node clusters as the ones that are associated the most with corresponding metastable states. Table 4.2 demonstrates a list of microbial species that are represented by above mentioned associated nodes. We believe that these results might provide new knowledge into which species or interactions of species might be responsible for or affected by the changes that the biological interaction network undergoes from the period of cholera infection to a period of recovery.

To gain a better understanding of the biological significance of these interpretations, it is necessary to delve deeper into the results. For example, previous studies (Langdon et al., 2016) have extensively focused on statistical analysis to determine the bacteria or species that are influenced by or influence shifts in microbiome compositions. By leveraging the identified species from these studies, we can compare them with the nodes that have exhibited the most significant impact on the model's output, as determined by the LRP approach. Furthermore, from Table 4.2 we see that a lot of species that are associated with the period of cholera infection are *Proteobacteria*. This result is supported by recent funding that *Proteobacteria* is a possible microbial signature of intestinal diseases (Rizzatti et al., 2017). However, we strongly believe these results must be explored further in depth.

4.4 Discussion

This chapter introduced a novel approach for extracting important and relevant microbial biomarkers associated with transitions of microbiome composition from a stable state to an alternative state. This knowledge can contribute to a com-

prehensive understanding of how the microbiome contributes to human health and well-being. Ultimately, we were able to answer a fundamental question about which species or interactions of species are responsible for or affected by changes in the biological interaction network from a healthy state to a diseased or antibiotic-exposed state. The proposed method provides not only valuable biological insights but also can serve as a method to validate the reliability of our proposed methods in Chapter 2 and Chapter 3 which are developed to simplify the analysis of microbial dynamics. Furthermore, the development of this approach is motivated by the challenges presented in the analysis of microbiome dynamics (discussed in Section 1.1), namely, emphasizing the importance of feature importance for identifying microbial biomarkers associated with changes in microbial composition.

In this chapter, we aimed to extend the methods proposed in Chapter 2 and Chapter 3. The underlying idea behind our approach is to simplify the extraction of microbiome dynamics by projecting it onto a low-dimensional space. This allows for the analysis of dynamics in this reduced space, avoiding the complexities of high-dimensional data. However, when we reduce the dimensionality of graph-structured data, we lose important information such as graph structural features (node or edge clusters, walks) that pertain to metastable states. The first method, graphKKE, is based on graph kernels that utilize feature mapping that cannot be explicitly defined. Consequently, it is impossible to reverse the process from the low-dimensional space back to the space of the time-evolving graph and extract structural features associated with metastable states. To address this limitation, we will employ a simple averaging method over time-snapshot graphs belonging to one metastable state. Since we have a finite set of metastable states, and according to the definition of metastability, the graphs within one metastable state should exhibit similarities in structure. While this simple method may not uncover more complex structural patterns, such as walks, we leave the exploration of such patterns for future work.

The second proposed method is a deep-learning method and they are often criticized for being "black boxes," making it difficult to interpret their results. Therefore, many interpretability methods have been proposed to gain insights into the decision-making processes of these models and which extend well beyond the mere interpretation of the model's results. While the primary objective of interpretability in machine learning is to understand the internal workings of the model and the factors contributing to its output, we leverage interpretability methods as an approach to identify the important biological components of the microbiome. This

enhances our understanding of which species of the human microbiome are responsible for transitioning from a stable state (as discussed in Chapter 1) to an alternative state under the influence of different perturbations, such as diseases or antibiotic exposure. Thus, the results produced in this chapter are mainly related to the extension of the deep-learning-based method.

We first provided an overview of explanation methods that utilize different concepts and explain different levels of deep learning-based models. These methods aim to provide explanations at either the individual feature level or provide insights into the overall or average behaviour of the model. The methods that we propose to use to extract relevant graph features belong to the first group of methods. The first method utilizes the attention matrix and the second method is based on layer-wise relevance propagation, a state-of-the-art explanation method. There are numerous studies showing that the attention weights do not contain the necessary information for the explanation of the deep learning model. On the other hand, there are studies that argue the statement mentioned above. We utilize this method for our purposes for the sake of experiments and comparison. We consider the main result to be from the LRP-based approach. We introduced several changes to the original LRP-based method in that we propagated relevance score not from the classification layer, but from the project head space. This is done due to the fact that most explanation methods focus on the explanation of the classification models, where the last layer outputs a class/label. In our case, we work with contrastive learning which implies that the output of the last layer indicates how similar two time-snapshot graphs are.

Finally, we conducted experiments on both synthetic and real-world data. In the case of synthetic data, we have ground-truth graph features which we defined during the construction step. Therefore, the experiments on synthetic data aim to understand the decision-making of the model and how reliable it is. We assume that if the model uses human-defined graph features, which are unique for each metastable state, then the low-dimensional vector produced by the model is reliable. Our experiments on the synthetic data have shown that the model proposed in Chapter 3 uses the relevant graph features, in this case, node clusters. Moreover, the findings obtained from both the attention weights and the LRP-based method align with each other. In terms of the real-world microbiome data, we do not have any ground-truth graph features that are associated with each metastable state. Therefore, we consider the results obtained from the experiments on real-world microbiome data to be novel. In the experiments, we also compared the attention-based method, the LRP-

based method, and the averaging approach for graphKKE. Each of these methods produced node clusters that are potentially associated with the time-evolving graph transitioning from the period of cholera infection to the period of recovery. Since the averaging approach for graphKKE showed the presence of two clear node clusters that are unique for two metastable states based on the average degree matrix, we did not consider the results of this method as final. Moreover, for the LRP-based method, we listed the names of microbial species that are associated with each metastable state. These results can be further explored in terms of biological meaning. In order to enhance our comprehension of the biological importance of these interpretations, it is crucial to delve further into the outcomes. For instance, we can utilize previous studies that identify bacteria or species affected by or influencing changes in microbiome compositions and compare them with the results obtained in this chapter.

Chapter 5

Discussion, Conclusion, and Outlook

It is a well-known fact that the microbiome has a strong influence on both its host and environment. Recent studies show large-scale perturbations in the microbiome constitution are heavily correlated, whether as a driver or a consequence, with different clinical diseases. Therefore, the analysis of microbiome data is crucial for gaining a better understanding of how the microbiome responds to these perturbations. This thesis introduced a new paradigm in the field of the analysis of microbiome dynamics and the unsupervised methods that facilitate this analysis.

5.1 Discussion and Conclusion

The high dimensionality and complexity of microbiome data are challenging to analyze with conventional methods due to the curse of dimensionality, overfitting, and various limitations that those approaches have. Furthermore, existing methods for analysing microbiome data often ignore either temporal changes or multiple interactions between species. Therefore, the first contribution of this thesis to the field is modelling the microbiome as a set of correlation networks constructed at each time step that allows analysis of temporal changes and identifying the species or interactions associated with the fluctuations of microbiome composition. The second contribution of the thesis is that we provide a novel method for analysing the microbiome in a lower-dimensional space while retaining its metastable behaviour and complex interactions. Analyzing the microbiome's dynamical properties in this new space is a step beyond the typical approach, which analyses the microbiome in high-dimensional space, thereby missing out on the critical information our method retains. This however requires an approach that can capture simultaneously both temporal and topological patterns of time-evolving networks and learn a low-dimensional representation simultaneously. This thesis has presented two approaches that address the above-mentioned objectives. Another important question that we aimed to answer in this thesis is the extraction of important microbial biomarkers that are associated with transitions of microbiome compositions from a stable state to an alternative state under the influence of different perturbations. Therefore, the final contribution of this thesis is the extension of these two approaches with their ability to obtain these important microbial biomarkers. Overall, we believe that four research questions defined in Chapter 1 have been answered by this thesis.

In Chapter 1 we discussed the importance of the analysis of microbiome data

and that recent studies have shown that microbiomes are not static over time. The main challenges of microbiome data, such as the curse of dimensionality, sparsity, compositionally, and feature importance, have been discussed. We presented common approaches for the analysis of microbiome data along with their limitations. Moreover, we discussed the current approaches for modeling the microbiome as a network to account for multiple complex interactions between species.

Chapter 2 presents the first method for learning a low-dimensional representation of a time-evolving graph. To capture the temporal changes in the time-evolving graph we propose using transfer operator theory. The idea is that spectral properties of transfer operators contain information about the global dynamical properties of the system. However, since the transfer operators are infinite-dimensional linear operators, it requires their approximation in a finite-dimensional space. Several data-driven methods have been proposed for which it is necessary to have a kernel-based estimation. Therefore, our method approximates transfer operators with graph kernels. Using graph kernels allows us to transform graphs into feature vectors without explicitly constructing high-dimensional feature space. We have also presented synthetic and real-world datasets, which have been used in further chapters throughout the thesis. The synthetic dataset is based on energy potential which introduces metastable states into a time-evolving graph. The time-evolving graph is constructed using $1d$ and $2d$ realizations of stochastic differential equations with corresponding potential or energy functions. For real-world data, we used two well-known microbiome datasets. The first one presents microbial compositions from different body parts. We added an additional Gaussian noise to the data in order to have metastable behaviour. The second dataset is the cholera infection of one individual. We used the Pearson correlation coefficient to construct an initial correlation network, which is not the best way to construct a network from a microbiome as we know from Chapter 1, since it does not address the compositionality of the microbiome. We conducted several experiments with both synthetic and real-world datasets. We projected the time-evolving graphs to a lower-dimensional space using the presented method. The k dominant eigenvalues of approximated transfer operators indicate the presence of k metastable states in the time-evolving graph. The resulting low-dimensional vector is represented by the eigenfunctions corresponding to this k dominant eigenvalues of transfer operators approximated with the proposed method. In the next experiment, which is comparative analysis, we have shown how this low-dimensional vector can be used for further analysis. For instance, the low-dimensional vector can be clustered

to identify the location of each metastable state. The vector can also be used with other machine learning methods, transition path theory, etc. In the case of microbiome data, we were able to identify a period of cholera infection and a period of recovery. Overall, we have shown that the combination of graph kernels and transfer operators allows us to extract important information about microbiome dynamics and to produce a low-dimensional vector that can be used for further analysis of microbiome dynamics.

In Chapter 3 we continue exploring how to embed a time-evolving graph onto a lower dimensional space while maintaining original dynamics. In particular, we studied how deep learning techniques can help to study the complex network of microbial species. We suggest that the proposed deep learning method is not only able to learn a low-dimensional representation but also we can combine it with interpretability methods to extract important graph features associated with microbiome composition transitioning from one state to another. This is the main advantage of this method over the method proposed in Chapter 2. We incorporated two main components in the proposed method: the state-of-the-art model for sequential data, Transformer, and contrastive learning. The Transformer architecture is widely used in large language models and has proven to be the most powerful class of architectures invented to date. Since we do not have ground-truth labels in our problem, we leveraged the unsupervised method, the so-called contrastive learning. It uses the principle of contrasting samples against each other to learn common attributes between time-snapshot graphs. One of the challenges in contrastive learning is that we need to define positive and negative pairs. We discussed different contrastive losses that employ various strategies for defining positive and negative pairs. For our problem, we based contrastive learning on the assumption that the current time-snapshot graph is the most similar to the consecutive time-snapshot graph (the definition of metastability). Furthermore, most of the graph representation learning methods focus on learning node embeddings, we, however, aim at obtaining a low-dimensional representation of the whole time-snapshot graph. Therefore, we integrated a so-called master node, which is connected to all other nodes in each time-snapshot graph. After the training, the master node at each time point is considered a low-dimensional representation of a corresponding time-snapshot graph. This Chapter also extends the synthetic data by proposing a time-evolving graph with different structural features. By doing so, we want to show that for our problem it is important to have positional information incorporated in the model to differ-

entiate graph structural features associated with metastable states. We presented a set of experiments that include the visual assessment of a low-dimensional representation as well as a quantitative assessment of the performance of the proposed method. We showed that during the training the model was able to detect different numbers of metastable states in the time-evolving graph. Since the proposed synthetic datasets are based on the realization of stochastic differentiated equations with a potential/energy function, we compared the initial trajectories that were used to construct the time-evolving graphs and the resulting low-dimensional representations. The results indicated that trajectories match almost identically. The quantitative assessment showed that our method outperforms other dimensionality reduction and graph representation learning methods in the clustering task. We also used the same real-world datasets to demonstrate that the proposed method was able to learn a low-dimensional representation while maintaining the original metastable behaviour. However, in the case of the real-world data the training was not stable due to the size of datasets. Transfer learning might be a solution that we leave for future work.

The main advantage of the method proposed in Chapter 2 is that we can identify the dimensionality of a new space with k dominant eigenvalues of approximated transfer operators close to one. In the deep learning method in Chapter 3, in turn, the number of dimensionality is a hyperparameter, which can influence the final result. Moreover, another advantage of the graphKKE method is that the number of metastable states in the time-evolving graph equals the number of dominant eigenvalues close to one. The deep learning-based method is, in turn, similar to any clustering method where the number of clusters has to be chosen. Yet, the deep learning-based method allows us to extract relevant graph structural features that are associated with metastable states, while in the case of graphKKE we define a feature space implicitly and we cannot extract relevant graph features. Chapter 4 was dedicated to exploring the possibility of extracting graph features in more detail. To the best of our knowledge, this chapter suggested a novel model in the microbiome field for extracting important microbial biomarkers associated with transitions of microbiome composition from a stable state to an alternative state under the influence of various perturbations. We proposed to use two different explanation approaches as a way of extracting associated graph features. Originally, the explanation methods are proposed to explain "black-box" machine learning models in order to prove that these models are reliable and trustworthy. In our case, we use it to ad-

dress one of the challenges in the microbiome analysis, namely feature importance. The first approach is based on attention weights. Since the deep learning-based method utilizes the attention mechanism, we extract attention weights to visualize the important graph features such as node clusters. Another approach is layer-wise relevance propagation, the state-of-the-art explanation technique, that allows extracting important features of the input space that help the model make predictions. We conducted experiments on both synthetic and microbiome data. We visualized the important and relevant graph structural features. In terms of microbiome data, we presented a table of species that according to our method are associated with the microbiome composition transitioning from the period of cholera infection to the period of recovery. We think that we obtained novel results which help to reveal underlying disease patterns in the data. Further study of these results is needed to investigate if these interpretations have biological meaning and the alignment of these results with results from previous studies.

We believe that the proposed methods in this thesis enhance the analysis of microbiome data by taking into account a huge amount of interaction between species and complex microbiome dynamics. On top of these methods, we have shown that by using various explanation techniques we can find what makes the model arrive at a certain low-dimensional representation. In terms of microbiome data, this means that our method, coupled with a proper interpretation strategy, can help reveal underlying disease patterns in the data.

5.2 Future Direction

Finally, we suggest potential future research directions that are related to the analysis of microbiome dynamics. We split the future directions into two parts: the one that explores future work in connection with the proposed method and another one that explores future directions in terms of microbiome data and network construction from the microbiome data.

Directed and attributed time-evolving graphs. Our experiments have included only an undirected time-evolving graph without any node and edge attributes. Yet, in real-world settings, especially in the microbiome community, there might be additional information about species and interactions that can be included as attributes to time-evolving graphs. Moreover, the dynamics of competitive and cooperat-

ive relationships between species can be modelled as a directed time-evolving graph, which poses an additional challenge to the analysis of microbiome dynamics. Therefore, future work may include an investigation of how well the proposed methods are capable of learning a low-dimensional representation of directed time-evolving graphs.

Transfer learning for real-world microbiome data. In our experiments, we have trained the model on small microbiome datasets, and in general, it is usually challenging to collect large datasets in biology. Training the model on a small dataset can result in overfitting and bad generalization. Transfer learning and fine-tuning can help to overcome this problem by leveraging knowledge gained from solving one problem and applying it to a different but related problem. In transfer learning, we can train a model on synthetic data and then use this pre-trained model on the small microbiome dataset.

Construction of a time-evolving graph from the microbiome data. How to construct a time-evolving graph from the microbiome data is a research question in itself. The thesis has not addressed this question which, therefore, might be a future direction. In this thesis, the Pearson correlation coefficient was used as a method to construct an initial correlation graph, which is not robust to the compositionality of the microbiome data (discussed in Chapter 1). We explored other existing approaches for constructing networks that address the limitation of the Pearson and Spearman correlation coefficient and can be used to construct an initial correlation graph. Furthermore, future work might also include the construction of a time-snapshot graph at each time point. In this thesis, we used a very simple approach in which we removed an edge if the concentration of a particular species was close to zero. However, this information had been already used when constructing the initial correlation matrix which might bias the results.

Thorough biological analysis of the results obtained from the interpretability of the model. The results presented in Chapter 4 lack a biological interpretation. We believe that this is an essential step for obtaining a better understanding of whether the proposed methods are reliable. This is especially important in the explanation method that reveals relevant graph features associated with metastable states. For instance, the study from (Hsiao et al., 2014) presents an extensive over-

view of bacterial taxa that are involved in recovery from cholera infection. So, the knowledge presented in this study can be compared with our results.

Reveling more complex graph features associated with metastable states. So far we have considered synthetic time-evolving graphs only with simple graph features associated with metastable states and in Chapter 4 we extracted simple graph structures from the microbiome dataset such as node and edge clusters. Future work might include learning a low-dimensional representation of a time-evolving graph with more complex graph structural features such as walks and the extraction of the explanation using methods discussed in the previous Chapter. Walks represent sequences of nodes or edges in a graph and can capture higher-order dependencies and temporal patterns within the graph. Instead of focusing only on pairwise interactions of species in the microbiome, extracting walks can shed light on more complex interactions between species

Bibliography

Ali, Ameen et al. (2022). 'XAI for transformers: Better explanations through conservative propagation'. In: *International Conference on Machine Learning*. PMLR, pp. 435–451.

Armstrong, George et al. (2022). 'Applications and comparison of dimensionality reduction methods for microbiome data'. In: *Frontiers in Bioinformatics* 2.

Arras, Leila et al. (2019). 'Explaining and interpreting LSTMs'. In: *Explainable ai: Interpreting, explaining and visualizing deep learning*, pp. 211–238.

Arslan, Salim et al. (2018). 'Graph saliency maps through spectral convolutional networks: Application to sex classification with brain connectivity'. In: *Graphs in Biomedical Image Analysis and Integrating Medical Imaging and Non-Imaging Modalities: Second International Workshop, GRAIL 2018 and First International Workshop, Beyond MIC 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 20, 2018, Proceedings 2*. Springer, pp. 3–13.

Asgari, Ehsaneddin et al. (2018). 'MicroPheno: predicting environments and host phenotypes from 16S rRNA gene sequencing using a k-mer based representation of shallow sub-samples'. In: *Bioinformatics* 34.13, pp. i32–i42.

Ba, Jimmy Lei, Jamie Ryan Kiros and Geoffrey E Hinton (2016). 'Layer normalization'. In: *arXiv preprint arXiv:1607.06450*.

Bach, Sebastian et al. (2015). 'On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation'. In: *PloS one* 10.7, e0130140.

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

Bai, Bing et al. (2021). 'Why attentions may not be interpretable?' In: *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 25–34.

Bair, Eric et al. (2006). 'Prediction by supervised principal components'. In: *Journal of the American Statistical Association* 101.473, pp. 119–137.

Bali, Prerna et al. (2021). 'Microbiome signatures in a fast-and slow-progressing gastric cancer murine model and their contribution to gastric carcinogenesis'. In: *Microorganisms* 9.1, p. 189.

Barros, Claudio D. T. et al. (2021). 'A Survey on Embedding Dynamic Graphs'. In: *ArXiv* abs/2101.01229.

Beauchamp-Walters, Julia et al. (2023). 'Impact of exclusive enteral nutrition on the gut microbiome of children with medical complexity'. In: *Journal of Parenteral and Enteral Nutrition* 47.1, pp. 77–86.

Borgwardt, Karsten M et al. (2005). 'Protein function prediction via graph kernels'. In: *Bioinformatics* 21.suppl_1, pp. i47–i56.

Bovier, A (2006). 'Metastability: a potential theoretic approach'. In: *Proceedings of the International Congress of Mathematicians*, 499–518.

Brauwers, Gianni and Flavius Frasincar (2021). 'A general survey on attention mechanisms in deep learning'. In: *IEEE Transactions on Knowledge and Data Engineering*.

Bruna, Joan et al. (2013). 'Spectral networks and locally connected networks on graphs'. In: *arXiv preprint arXiv:1312.6203*.

Campbell, Tayte P et al. (2020). 'The microbiome and resistome of chimpanzees, gorillas, and humans across host lifestyle and geography'. In: *The ISME journal* 14.6, pp. 1584–1599.

Candès, Emmanuel J et al. (2011). 'Robust principal component analysis?' In: *Journal of the ACM (JACM)* 58.3, pp. 1–37.

Caporaso, J.G. et al. (May 2011). 'Moving pictures of the human microbiome'. In: *Genome biology* 12, R50.

Carrillo, Alfredo, Luis F Cantú and Alejandro Noriega (2021). 'Individual Explanations in Machine Learning Models: A Survey for Practitioners'. In: *arXiv preprint arXiv:2104.04144*.

Chaddad, Ahmad et al. (2023). 'Survey of Explainable AI Techniques in Healthcare'. In: *Sensors* 23.2, p. 634.

Chang, William K., David VanInsberghe and Libusha Kelly (2020). 'Topological analysis reveals state transitions in human gut and marine bacterial communities'. In: *NPJ Biofilms and Microbiomes* 6.

Chefer, Hila, Shir Gur and Lior Wolf (2021a). 'Generic attention-model explainability for interpreting bi-modal and encoder-decoder transformers'. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 397–406.

— (2021b). 'Transformer interpretability beyond attention visualization'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 782–791.

Chen, Ting et al. (2020). 'A simple framework for contrastive learning of visual representations'. In: *International conference on machine learning*. PMLR, pp. 1597–1607.

Cho, Kyunghyun et al. (2014). 'Learning phrase representations using RNN encoder-decoder for statistical machine translation'. In: *arXiv preprint arXiv:1406.1078*.

Chopra, Sumit, Raia Hadsell and Yann LeCun (2005). 'Learning a similarity metric discriminatively, with application to face verification'. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. Vol. 1. IEEE, pp. 539–546.

Choromanski, Krzysztof et al. (2020). 'Rethinking attention with performers'. In: *arXiv preprint arXiv:2009.14794*.

Clark, Kevin et al. (2019). 'What does bert look at? an analysis of bert's attention'. In: *arXiv preprint arXiv:1906.04341*.

Clarke, K Robert (1993). 'Non-parametric multivariate analyses of changes in community structure'. In: *Australian journal of ecology* 18.1, pp. 117–143.

Craven, Mark and Jude Shavlik (1995). 'Extracting tree-structured representations of trained networks'. In: *Advances in neural information processing systems* 8.

Cui, Peng et al. (2019). 'A Survey on Network Embedding'. In: *IEEE Transactions on Knowledge and Data Engineering* 31, pp. 833–852.

Danel, Tomasz et al. (2020). 'Spatial graph convolutional networks'. In: *Neural Information Processing: 27th International Conference, ICONIP 2020, Bangkok, Thailand, November 18–22, 2020, Proceedings, Part V*. Springer, pp. 668–675.

David, Lawrence et al. (July 2014). 'Host lifestyle affects human microbiota on daily timescales'. In: *Genome biology* 15, R89.

De Winter, Sam et al. (2018). 'Combining temporal aspects of dynamic networks with Node2Vec for a more efficient dynamic link prediction'. In: *2018 IEEE/ACM international conference on advances in social networks analysis and mining (ASONAM)*. IEEE, pp. 1234–1241.

Deng, Ye et al. (2012). 'Molecular ecological network analyses'. In: *BMC bioinformatics* 13, pp. 1–20.

Dethlefsen, Les and David A Relman (2011). 'Incomplete recovery and individualized responses of the human distal gut microbiota to repeated antibiotic perturbation'. In: *Proceedings of the National Academy of Sciences* 108.supplement_1, pp. 4554–4561.

Devlin, Jacob et al. (June 2019). 'BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding'. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 4171–4186. URL: https://aclanthology.org/N19-1423.

Doshi-Velez, Finale and Been Kim (2017). 'Towards a rigorous science of interpretable machine learning'. In: *arXiv preprint arXiv:1702.08608*.

Dosovitskiy, Alexey et al. (2020). 'An image is worth 16x16 words: Transformers for image recognition at scale'. In: *arXiv preprint arXiv:2010.11929*.

Durack, J and S V Lynch (2019). 'The gut microbiome: Relationships with disease and opportunities for therapy'. In: *The Journal of experimental medicine* 216.1, pp. 20–40.

Dwivedi, Vijay Prakash and Xavier Bresson (2020). 'A generalization of transformer networks to graphs'. In: *arXiv preprint arXiv:2012.09699*.

Eberle, Oliver et al. (2020). 'Building and interpreting deep similarity models'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.3, pp. 1149–1161.

Epasto, Alessandro and Bryan Perozzi (2019). 'Is a single embedding enough? learning node representations that capture multiple social contexts'. In: *The world wide web conference*, pp. 394–404.

Faith, Jeremiah J. et al. (2013). 'The Long-Term Stability of the Human Gut Microbiota'. In: *Science* 341.6141, p. 1237439. eprint: https://www.science.org/doi/pdf/10.1126/science.1237439. URL: https://www.science.org/doi/abs/10.1126/science.1237439.

Fang, Huaying et al. (2015). 'CCLasso: correlation inference for compositional data through Lasso'. In: *Bioinformatics* 31.19, pp. 3172–3180.

Faust, Karoline and Jeroen Raes (2012). 'Microbial interactions: from networks to models'. In: *Nature Reviews Microbiology* 10.8, pp. 538–550.

Fedus, William, Barret Zoph and Noam Shazeer (2022). 'Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity'. In: *The Journal of Machine Learning Research* 23.1, pp. 5232–5270.

Feng, Aosong et al. (2022). 'Kergnns: Interpretable graph neural networks with graph kernels'. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 36. 6, pp. 6614–6622.

Feragen, Aasa et al. (2013). 'Scalable kernels for graphs with continuous attributes'. In: *Advances in neural information processing systems* 26.

Fisher, Charles and Pankaj Mehta (Feb. 2014). 'Identifying Keystone Species in the Human Gut Microbiome from Metagenomic Timeseries Using Sparse Linear Regression'. In: *PloS one* 9.

Friedman, Jonathan and Eric J Alm (2012). 'Inferring correlation networks from genomic survey data'. In: *PLoS computational biology* 8.9, e1002687.

Frosst, Nicholas, Nicolas Papernot and Geoffrey Hinton (2019). 'Analyzing and improving representations with the soft nearest neighbor loss'. In: *International conference on machine learning*. PMLR, pp. 2012–2020.

Gajer, Pawel et al. (2012). 'Temporal dynamics of the human vaginal microbiota'. In: *Science translational medicine* 4.132, 132ra52–132ra52.

Gärtner, T, PA Flach and S Wrobel (2003a). 'On graph kernels: Hardness results and efficient alternatives.' In: *Lecture Notes in Computer Science*, pp. 129–143.

Gärtner, Thomas, Peter Flach and Stefan Wrobel (2003b). 'On graph kernels: Hardness results and efficient alternatives'. In: *Learning Theory and Kernel Machines: 16th Annual Conference on Learning Theory and 7th Kernel Workshop, COLT/Kernel 2003, Washington, DC, USA, August 24-27, 2003. Proceedings*. Springer, pp. 129–143.

Gerber, Georg K (2014). 'The dynamic microbiome'. In: *FEBS letters* 588.22, pp. 4131–4139.

Gilmer, Justin et al. (2017). 'Neural message passing for quantum chemistry'. In: *International conference on machine learning*. PMLR, pp. 1263–1272.

Gloor G. B., Macklaim J. M. Egozcue J. J. (2017). 'Microbiome Datasets Are Compositional: And This Is Not Optional.' In: *Frontiers in Microbiology*.

Gonze, Didier et al. (2018). 'Microbial communities as dynamical systems'. In: *Current opinion in microbiology* 44, pp. 41–49.

Gopalakrishnan, V et al. (2018). 'The influence of the gut microbiome on cancer, immunity, and cancer immunotherapy'. In: *Cancer Cell* 33.4, pp. 570–580.

Goyal, P, S Rokka Chhetri and A Canedo (2020). 'dyngraph2vec: Capturing Network Dynamics using Dynamic Graph Representation Learning'. In: *Knowl. Based Syst.* 187.

Goyal, Palash et al. (2018). 'DynGEM: Deep Embedding Method for Dynamic Graphs'. In: *ArXiv* abs/1805.11273.

Grover, A and J Leskovec (2016). 'Node2vec: Scalable Feature Learning for Networks'. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 855–864.

Gutmann, Michael and Aapo Hyvärinen (2010). 'Noise-contrastive estimation: A new estimation principle for unnormalized statistical models'. In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Yee Whye Teh and Mike Titterington. Vol. 9. Proceedings of Machine Learning Research. Chia Laguna Resort, Sardinia, Italy: PMLR, pp. 297–304. URL: https://proceedings.mlr.press/v9/gutmann10a.html.

Hadsell, Raia, Sumit Chopra and Yann LeCun (2006). 'Dimensionality reduction by learning an invariant mapping'. In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*. Vol. 2. IEEE, pp. 1735–1742.

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

Hanif, Ambreen, Xuyun Zhang and Steven Wood (2021). 'A survey on explainable artificial intelligence techniques and challenges'. In: *2021 IEEE 25th international enterprise distributed object computing workshop (EDOCW)*. IEEE, pp. 81–89.

Hao, Yaru et al. (2021). 'Self-attention attribution: Interpreting information interactions inside transformer'. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 14, pp. 12963–12971.

Harchaoui, Zaïd and Francis Bach (2007). 'Image classification with segmentation graph kernels'. In: *2007 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 1–8.

He, Kaiming et al. (2016). 'Deep residual learning for image recognition'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

He, Mingguo, Zhewei Wei and Ji-Rong Wen (2022). 'Convolutional neural networks on graphs with chebyshev approximation, revisited'. In: *arXiv preprint arXiv:2202.03580*.

Heidari, Farzaneh and Manos Papagelis (2020). 'Evolving network representation learning based on random walks'. In: *Applied network science* 5, pp. 1–38.

Hernández-Rocha, Cristian et al. (2021). 'Integrative analysis of colonic biopsies from inflammatory bowel disease patients identifies an interaction between microbial bile acid-inducible gene abundance and human angiopoietin-like 4 gene expression'. In: *Journal of Crohn's and Colitis* 15.12, pp. 2078–2087.

Hjorth, MF et al. (Mar. 2018). 'Pre-treatment microbial Prevotella-to-Bacteroides ratio, determines body fat loss success during a 6-month randomized controlled diet intervention'. In: *International Journal of Obesity* 42.3, pp. 580–583.

Hoyer, Patrik O (2004). 'Non-negative matrix factorization with sparseness constraints.' In: *Journal of machine learning research* 5.9.

Hsiao, A et al. (2014). 'Members of the human gut microbiota involved in recovery from Vibrio cholerae infection'. In: *Nature* 515, pp. 423–426.

Islam, Sheikh Rabiul et al. (2021). 'Explainable artificial intelligence approaches: A survey'. In: *arXiv preprint arXiv:2101.09429*.

Jacobs, Robert A et al. (1991). 'Adaptive mixtures of local experts'. In: *Neural computation* 3.1, pp. 79–87.

Jain, Sarthak and Byron C Wallace (2019). 'Attention is not explanation'. In: *arXiv preprint arXiv:1902.10186*.

Joseph, Tyler A et al. (2020). 'Compositional Lotka-Volterra describes microbial dynamics in the simplex'. In: *PLOS Computational Biology* 16.5, e1007917.

Kashima, Hisashi, Koji Tsuda and Akihiro Inokuchi (2003). 'Marginalized kernels between labeled graphs'. In: *Proceedings of the 20th international conference on machine learning (ICML-03)*, pp. 321–328.

Kazemi, Seyed Mehran et al. (2020). 'Representation Learning for Dynamic Graphs: A Survey'. In: *J. Mach. Learn. Res.* 21, 70:1–70:73.

Khoshraftar, Shima and Aijun An (2022). *A Survey on Graph Representation Learning Methods*. URL: https://arxiv.org/abs/2204.01855.

Khoshraftar, Shima et al. (2019). 'Dynamic graph embedding via LSTM History tracking'. In: *2019 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE, pp. 119–127.

Kim, Been et al. (2018). 'Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav)'. In: *International conference on machine learning*. PMLR, pp. 2668–2677.

Kincaid, HJ, R Nagpal and H Yadav (2019). 'Microbiome-immune-metabolic axis in the epidemic of childhood obesity: Evidence and opportunities'. In: *Obesity Reviews* 21.2.

Kipf, Thomas N and Max Welling (2016a). 'Semi-supervised classification with graph convolutional networks'. In: *arXiv preprint arXiv:1609.02907*.

— (2016b). 'Variational graph auto-encoders'. In: *arXiv preprint arXiv:1611.07308*.

Klus, S, P Koltai and C Schütte (Sept. 2016). 'On the numerical approximation of the Perron–Frobenius and Koopman operator'. In: *Journal of Computational Dynamics* 3, pp. 51–79.

Klus, S, I Schuster and K Muandet (2019a). 'Eigendecompositions of Transfer Operators in Reproducing Kernel Hilbert Spaces'. In: *Journal of Nonlinear Science* 30, 283–315.

Klus, S et al. (2019b). 'Kernel methods for detecting coherent structures in dynamical data'. In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 29.12.

Klus, Stefan et al. (2018a). 'A kernel-based approach to molecular conformation analysis'. In: *The Journal of Chemical Physics* 149.24, p. 244109.

Klus, Stefan et al. (2018b). 'Data-driven model reduction and transfer operator approximation'. In: *Journal of Nonlinear Science* 28, pp. 985–1010.

Kohlbrenner, Maximilian et al. (2020). 'Towards best practice in explaining neural network decisions with LRP'. In: *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, pp. 1–7.

Kondor, Risi and Horace Pan (2016). 'The multiscale laplacian graph kernel'. In: *Advances in neural information processing systems* 29.

Kreuzer, Devin et al. (2021). 'Rethinking graph transformers with spectral attention'. In: *Advances in Neural Information Processing Systems* 34, pp. 21618–21629.

Kriege, Nils and Petra Mutzel (2012). 'Subgraph matching kernels for attributed graphs'. In: *arXiv preprint arXiv:1206.6483*.

Kriege, Nils M, Fredrik D Johansson and Christopher Morris (2020). 'A survey on graph kernels'. In: *Applied Network Science* 5.1, pp. 1–42.

Kriege, Nils Morten (2015). 'Comparing graphs'. In.

Lahti, Leo et al. (2014). 'Tipping elements in the human intestinal ecosystem'. In: *Nature Communications* 5.1. URL: https://doi.org/10.1038%2Fncomms5344.

Langdon, Amy Elizabeth, Nathan Crook and Gautam Dantas (2016). 'The effects of antibiotics on the microbiome throughout development and alternative approaches for therapeutic modulation'. In: *Genome Medicine* 8.

Lawrence, Neil and Aapo Hyvärinen (2005). 'Probabilistic non-linear principal component analysis with Gaussian process latent variable models.' In: *Journal of machine learning research* 6.11.

Layeghifard, Mehdi, David M Hwang and David S Guttman (2017). 'Disentangling interactions in the microbiome: a network perspective'. In: *Trends in microbiology* 25.3, pp. 217–228.

Lee, Daniel D and H Sebastian Seung (1999). 'Learning the parts of objects by non-negative matrix factorization'. In: *Nature* 401.6755, pp. 788–791.

Lee, Junhyun, Inyeop Lee and Jaewoo Kang (2019). 'Self-Attention Graph Pooling'. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, pp. 3734–3743. URL: https://proceedings.mlr.press/v97/lee19c.html.

Leong, Claudia et al. (2020). 'Using compositional principal component analysis to describe children's gut microbiota in relation to diet and body composition'. In: *The American Journal of Clinical Nutrition* 111.1, pp. 70–78.

Li, Yaguang et al. (2017). 'Diffusion convolutional recurrent neural network: Data-driven traffic forecasting'. In: *arXiv preprint arXiv:1707.01926*.

Li, Yuan et al. (2019). 'Graph transformer'. In.

Li, Yujia et al. (2015). 'Gated graph sequence neural networks'. In: *arXiv preprint arXiv:1511.05493*.

Lin, Tianyang et al. (2022). 'A survey of transformers'. In: *AI Open*.

Liu, Yibing et al. (2022). 'Rethinking attention-model explainability through faithfulness violation test'. In: *International Conference on Machine Learning*. PMLR, pp. 13807–13824.

Lo, C and R Marculescu (2019). 'MetaNN: accurate classification of host phenotypes from metagenomic data using neural networks'. In: *BMC Bioinformatics* 20.314.

Long, Yahui et al. (2020). 'Ensembling graph attention networks for human microbe–drug association prediction'. In: *Bioinformatics* 36.Supplement_2, pp. i779–i786.

Lugo-Martinez, Jose et al. (2019). 'Dynamic interaction network inference from longitudinal microbiome data'. In: *Microbiome* 7, pp. 1–14.

Lundberg, Scott M and Su-In Lee (2017). 'A unified approach to interpreting model predictions'. In: *Advances in neural information processing systems* 30.

Luong, Minh-Thang, Hieu Pham and Christopher D Manning (2015). 'Effective approaches to attention-based neural machine translation'. In: *arXiv preprint arXiv:1508.04025*.

Maaten, Laurens Van der and Geoffrey Hinton (2008). 'Visualizing data using t-SNE.' In: *Journal of machine learning research* 9.11.

Mahdavi, Sedigheh, Shima Khoshraftar and Aijun An (2018). 'dynnode2vec: Scalable dynamic network embedding'. In: *2018 IEEE International Conference on Big Data (Big Data)*. IEEE, pp. 3762–3765.

Martino, Cameron et al. (2021). 'Context-aware dimensionality reduction deconvolutes gut microbial community dynamics'. In: *Nature biotechnology* 39.2, pp. 165–168.

McInnes, Leland, John Healy and James Melville (2018). 'Umap: Uniform manifold approximation and projection for dimension reduction'. In: *arXiv preprint arXiv:1802.03426*.

Melnyk, Kateryna, Kuba Weimann and Tim OF Conrad (2023). 'Understanding microbiome dynamics via interpretable graph representation learning'. In: *Scientific Reports* 13.1, p. 2058.

Melnyk, Kateryna et al. (2020). 'GraphKKE: graph Kernel Koopman embedding for human microbiome analysis'. In: *Applied Network Science* 5.1. URL: https://doi.org/10.1007%2Fs41109-020-00339-2.

Menni, C et al. (July 2017). 'Gut microbiome diversity and high-fibre intake are related to lower long-term weight gain'. In: *International Journal of Obesity* 41.7, pp. 1099–1105.

Micheli, Alessio, Diego Sona and Alessandro Sperduti (2004). 'Contextual processing of structured data by recursive cascade correlation'. In: *IEEE Transactions on Neural Networks* 15.6, pp. 1396–1410.

Mitrovic, Sandra and Jochen De Weerdt (2018). 'Dyn2Vec: Exploiting dynamic behaviour using difference networks-based node embeddings for classification'. In: *Proceedings of the International Conference on Data Science*, pp. 194–200.

Molgedey, Lutz and Heinz Georg Schuster (1994). 'Separation of a mixture of independent signals using time delayed correlations'. In: *Physical review letters* 72.23, p. 3634.

Momeni, Babak, Li Xie and Wenying Shou (2017). 'Lotka-Volterra pairwise modeling fails to capture diverse pairwise microbial interactions'. In: *eLife* 6. Ed. by Bruce Levin, e25051. ISSN: 2050-084X. URL: https://doi.org/10.7554/eLife.25051.

Montavon, Grégoire et al. (2017). 'Explaining nonlinear classification decisions with deep taylor decomposition'. In: *Pattern recognition* 65, pp. 211–222.

Montavon, Grégoire et al. (2019). 'Layer-wise relevance propagation: an overview'. In: *Explainable AI: interpreting, explaining and visualizing deep learning*, pp. 193–209.

Morris, Christopher et al. (2016). 'Faster kernels for graphs with continuous attributes via hashing'. In: *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, pp. 1095–1100.

Morris, Christopher et al. (2019). 'Weisfeiler and leman go neural: Higher-order graph neural networks'. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 33. 01, pp. 4602–4609.

Morris, Christopher et al. (2020). 'Tudataset: A collection of benchmark datasets for learning with graphs'. In: *arXiv preprint arXiv:2007.08663*.

Morton, James T et al. (2017). 'Uncovering the horseshoe effect in microbial analyses'. In: *Msystems* 2.1, e00166–16.

Morton, James T et al. (2019). 'Establishing microbial composition measurement standards with reference frames'. In: *Nature communications* 10.1, p. 2719.

Narayanan, Annamalai et al. (2017). 'graph2vec: Learning Distributed Representations of Graphs'. In: *ArXiv*.

Neumann, Marion et al. (2016). 'Propagation kernels: efficient graph kernels from propagated information'. In: *Machine Learning* 102, pp. 209–245.

Nikolentzos, Giannis, Giannis Siglidis and Michalis Vazirgiannis (2021). 'Graph kernels: A survey'. In: *Journal of Artificial Intelligence Research* 72, pp. 943–1027.

Nuske, Feliks et al. (2014). 'Variational approach to molecular kinetics'. In: *Journal of chemical theory and computation* 10.4, pp. 1739–1752.

Oh Song, Hyun et al. (2016). 'Deep metric learning via lifted structured feature embedding'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4004–4012.

Oord, Aaron van den, Yazhe Li and Oriol Vinyals (2018). 'Representation learning with contrastive predictive coding'. In: *arXiv preprint arXiv:1807.03748*.

Orsini, Francesco, Paolo Frasconi and Luc De Raedt (2015). 'Graph invariant kernels'. In: *Proceedings of the twenty-fourth international joint conference on artifi-*

cial intelligence. Vol. 2015. IJCAI-INT JOINT CONF ARTIF INTELL, pp. 3756–3762.

Parbie, Prince Kofi et al. (2021). 'Dysbiotic fecal microbiome in HIV-1 infected individuals in Ghana'. In: *Frontiers in Cellular and Infection Microbiology* 11, p. 646467.

Pareja, Aldo et al. (2019). *EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs*. URL: https://arxiv.org/abs/1902.10191.

Pérez-Cobas, Ana Elena et al. (2013). 'Gut microbiota disturbance during antibiotic therapy: a multi-omic approach'. In: *Gut* 62.11, pp. 1591–1601.

Perozzi, B, R Al-Rfou and S Skiena (2014). 'DeepWalk: Online Learning of Social Representations'. In: *KDD '14: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 701–710.

Pope, Phillip E et al. (2019). 'Explainability methods for graph convolutional neural networks'. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10772–10781.

Qin, J et al. (2012). 'A metagenome-wide association study of gut microbiota in type 2 diabetes'. In: *Nature* 490, pp. 55–60.

Ribeiro, Marco Tulio, Sameer Singh and Carlos Guestrin (2016). '" Why should i trust you?" Explaining the predictions of any classifier'. In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144.

Riera, Joan Lluís and Laura Baldo (2020). 'Microbial co-occurrence networks of gut microbiota reveal community conservation and diet-associated shifts in cichlid fishes'. In: *Animal Microbiome* 2, pp. 1–13.

Rizzatti, G et al. (2017). 'Proteobacteria: a common factor in human diseases'. In: *BioMed research international* 2017.

Robinson, Joshua et al. (2020). 'Contrastive learning with hard negative samples'. In: *arXiv preprint arXiv:2010.04592*.

RossD, LimJ, LinRS LimJ et al. (2008). 'Incrementallearningforrobustvisualtracking'. In: *InternationalJournalofComputerVision* 77.1r3, 125r141.

Roy, Aurko et al. (2021). 'Efficient content-based sparse attention with routing transformers'. In: *Transactions of the Association for Computational Linguistics* 9, pp. 53–68.

Saeed, Waddah and Christian Omlin (2023). 'Explainable ai (xai): A systematic meta-survey of current challenges and future opportunities'. In: *Knowledge-Based Systems*, p. 110273.

Sajjad, Hooman Peiro, Andrew Docherty and Yuriy Tyshetskiy (2019). 'Efficient representation learning using random walks for dynamic graphs'. In: *arXiv preprint arXiv:1901.01346*.

Samek, Wojciech et al. (2022). 'Explaining the Decisions of Convolutional and Recurrent Neural Networks'. In: *Mathematical Aspects of Deep Learning*, p. 229.

Sankar, Aravind et al. (2020). 'DySAT: Deep Neural Representation Learning on Dynamic Graphs via Self-Attention Networks'. In: *Proceedings of the 13th International Conference on Web Search and Data Mining*, pp. 519–527.

Schmid, Peter J (2010). 'Dynamic mode decomposition of numerical and experimental data'. In: *Journal of fluid mechanics* 656, pp. 5–28.

Schnake, Thomas et al. (2021). 'Higher-order explanations of graph neural networks via relevant walks'. In: *IEEE transactions on pattern analysis and machine intelligence* 44.11, pp. 7581–7596.

Schölkopf, Bernhard, Alexander Smola and Klaus-Robert Müller (2005). 'Kernel principal component analysis'. In: *Artificial Neural Networks—ICANN'97: 7th International Conference Lausanne, Switzerland, October 8–10, 1997 Proceedings*. Springer, pp. 583–588.

Schroff, Florian, Dmitry Kalenichenko and James Philbin (2015). 'FaceNet: A unified embedding for face recognition and clustering'. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. URL: https://doi.org/10.1109%2Fcvpr.2015.7298682.

Schulz, Till Hendrik et al. (2022). 'A generalized weisfeiler-lehman graph kernel'. In: *Machine Learning* 111.7, pp. 2601–2629.

Schwantes, Christian R and Vijay S Pande (2015). 'Modeling molecular kinetics with tICA and the kernel trick'. In: *Journal of chemical theory and computation* 11.2, pp. 600–608.

Selvaraju, Ramprasaath R et al. (2017). 'Grad-cam: Visual explanations from deep networks via gradient-based localization'. In: *Proceedings of the IEEE international conference on computer vision*, pp. 618–626.

Seo, Youngjoo et al. (2018). 'Structured sequence modeling with graph convolutional recurrent networks'. In: *Neural Information Processing: 25th International*

Conference, ICONIP 2018, Siem Reap, Cambodia, December 13-16, 2018, Proceedings, Part I 25. Springer, pp. 362–373.

Serrano, Sofia and Noah A Smith (2019). 'Is attention interpretable?' In: *arXiv preprint arXiv:1906.03731*.

Shaw, L et al. (Mar. 2019). 'Modelling microbiome recovery after antibiotics using a stability landscape framework'. In: *The ISME Journal* 13, pp. 1–12.

Shaw, Peter, Jakob Uszkoreit and Ashish Vaswani (2018). 'Self-attention with relative position representations'. In: *arXiv preprint arXiv:1803.02155*.

Shazeer, Noam et al. (2018). 'Mesh-tensorflow: Deep learning for supercomputers'. In: *Advances in neural information processing systems* 31.

Shen, Haipeng and Jianhua Z Huang (2008). 'Sparse principal component analysis via regularized low rank matrix approximation'. In: *Journal of multivariate analysis* 99.6, pp. 1015–1034.

Shen, Tao et al. (2018). 'Reinforced self-attention network: a hybrid of hard and soft attention for sequence modeling'. In: *arXiv preprint arXiv:1801.10296*.

Shervashidze, N et al. (2009). 'Efficient graphlet kernels for large graph comparison'. In: *Proceedings of Machine Learning Research*. Vol. 5, pp. 488–495.

Shervashidze, Nino et al. (2011). 'Weisfeiler-lehman graph kernels.' In: *Journal of Machine Learning Research* 12.9.

Shrikumar, Avanti, Peyton Greenside and Anshul Kundaje (2017). 'Learning important features through propagating activation differences'. In: *International conference on machine learning*. PMLR, pp. 3145–3153.

Simonyan, Karen, Andrea Vedaldi and Andrew Zisserman (2013). 'Deep inside convolutional networks: Visualising image classification models and saliency maps'. In: *arXiv preprint arXiv:1312.6034*.

Sohn, Kihyuk (2016). 'Improved deep metric learning with multi-class n-pair loss objective'. In: *Advances in neural information processing systems* 29.

Song, Eun-Ji, Eun-Sook Lee and Young-Do Nam (2018). 'Progress of analytical tools and techniques for human gut microbiome research'. In: *Journal of Microbiology* 56, pp. 693–705.

Sperduti, Alessandro and Antonina Starita (1997). 'Supervised neural networks for the classification of structures'. In: *IEEE Transactions on Neural Networks* 8.3, pp. 714–735.

Springenberg, Jost Tobias et al. (2014). 'Striving for simplicity: The all convolutional net'. In: *arXiv preprint arXiv:1412.6806*.

Stanaway, Ian B et al. (2023). 'Alteration of oral microbiome composition in children living with pesticide-exposed farm workers'. In: *International Journal of Hygiene and Environmental Health* 248, p. 114090.

Stein, Richard R et al. (2013). 'Ecological modeling from time-series inference: insight into dynamics and stability of intestinal microbiota'. In: *PLoS computational biology* 9.12, e1003388.

Sánchez-Alcoholado, L et al. (2020). 'The Role of the Gut Microbiome in Colorectal Cancer Development and Therapy Response'. In: *Cancers* 12.6.

Tabassum, Afrina et al. (2022). 'Hard negative sampling strategies for contrastive representation learning'. In: *arXiv preprint arXiv:2206.01197*.

Ter Braak, Cajo JF (1986). 'Canonical correspondence analysis: a new eigenvector technique for multivariate direct gradient analysis'. In: *Ecology* 67.5, pp. 1167–1179.

Tipping, Michael E and Christopher M Bishop (1999). 'Probabilistic principal component analysis'. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 61.3, pp. 611–622.

Van Der Maaten, Laurens (2009). 'Learning a parametric embedding by preserving local structure'. In: *Artificial intelligence and statistics*. PMLR, pp. 384–391.

— (2014). 'Accelerating t-SNE using tree-based algorithms'. In: *The journal of machine learning research* 15.1, pp. 3221–3245.

Vaswani, A. et al. (2017). 'Attention is All You Need'. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 6000–6010.

Velickovic, Petar et al. (2017). 'Graph attention networks'. In: *stat* 1050.20, pp. 10–48550.

Vishwanathan, S Vichy N et al. (2010). 'Graph kernels'. In: *Journal of Machine Learning Research* 11, pp. 1201–1242.

Wang, Daixin, Peng Cui and Wenwu Zhu (2016). 'Structural deep network embedding'. In: *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1225–1234.

Wang, Feng and Huaping Liu (2021). 'Understanding the behaviour of contrastive loss'. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2495–2504.

Wang, Shoujin et al. (2020a). 'Graph learning approaches to recommender systems: A review'. In: *arXiv preprint arXiv:2004.11718*.

Wang, Sinong et al. (2020b). 'Linformer: Self-attention with linear complexity'. In: *arXiv preprint arXiv:2006.04768*.

Wang, Xiang et al. (2019). 'Kgat: Knowledge graph attention network for recommendation'. In: *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 950–958.

Wiegreffe, Sarah and Yuval Pinter (2019). 'Attention is not not explanation'. In: *arXiv preprint arXiv:1908.04626*.

Williams, Matthew O, Ioannis G Kevrekidis and Clarence W Rowley (2015). 'A data–driven approximation of the koopman operator: Extending dynamic mode decomposition'. In: *Journal of Nonlinear Science* 25, pp. 1307–1346.

Williams, Ronald J (1992). 'Simple statistical gradient-following algorithms for connectionist reinforcement learning'. In: *Machine learning* 8.3, pp. 229–256.

Wright, John et al. (2009). 'Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization'. In: *Advances in neural information processing systems* 22.

Wu, Ziyou, Steven L Brunton and Shai Revzen (2021). 'Challenges in dynamic mode decomposition'. In: *Journal of the Royal Society Interface* 18.185, p. 20210686.

Wu, Zonghan et al. (2019). 'Graph wavenet for deep spatial-temporal graph modeling'. In: *arXiv preprint arXiv:1906.00121*.

Wu, Zonghan et al. (2020). 'A comprehensive survey on graph neural networks'. In: *IEEE transactions on neural networks and learning systems* 32.1, pp. 4–24.

Xiong, Ping et al. (2022). 'Efficient Higher-Order Subgraph Attribution via Message Passing'. PhD thesis. MA thesis. Technische Universität Berlin.

Xiong, Ping et al. (2023). 'Relevant Walk Search for Explaining Graph Neural Networks'. In.

Xu, Kelvin et al. (2015). 'Show, attend and tell: Neural image caption generation with visual attention'. In: *International conference on machine learning*. PMLR, pp. 2048–2057.

Xu, R and Q Wang (2016). 'Towards understanding brain-gut-microbiome connections in Alzheimer's disease.' In: *BMC Systems Biology* 10.63.

Xu, Xueli et al. (2020). 'A t-SNE based classification approach to compositional microbiome data'. In: *Frontiers in Genetics* 11, p. 620143.

Xuan, Hong et al. (2020). 'Hard negative examples are hard, but useful'. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*. Springer, pp. 126–142.

Yang, Chengliang, Anand Rangarajan and Sanjay Ranka (2018). 'Global model interpretation via recursive partitioning'. In: *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*. IEEE, pp. 1563–1570.

Ying, Rex et al. (2018a). 'Graph convolutional neural networks for web-scale recommender systems'. In: *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 974–983.

Ying, Zhitao et al. (2018b). 'Hierarchical Graph Representation Learning with Differentiable Pooling'. In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2018/file/e77dbaf6759253c7c6d0efc5690369c7-Paper.pdf.

Yu, Wenchao, Charu C Aggarwal and Wei Wang (2017). 'Temporally factorized network modeling for evolutionary network analysis'. In: *Proceedings of the Tenth ACM International conference on web search and data mining*, pp. 455–464.

Zackular, JP et al. (Nov. 2015). 'Manipulation of the Gut Microbiota Reveals Role in Colon Tumorigenesis'. In: *mSphere* 1.1.

Zaura, Egija et al. (2015). 'Same Exposure but Two Radically Different Responses to Antibiotics: Resilience of the Salivary Microbiome versus Long-Term Microbial Shifts in Feces'. In: *mBio* 6.

Zhang, Daokun et al. (2020a). 'Network Representation Learning: A Survey'. In: *IEEE Transactions on Big Data* 6, pp. 3–28.

Zhang, Jiani et al. (2018a). 'Gaan: Gated attention networks for learning on large and spatiotemporal graphs'. In: *arXiv preprint arXiv:1803.07294*.

Zhang, Jiawei et al. (2020b). 'Graph-bert: Only attention is needed for learning graph representations'. In: *arXiv preprint arXiv:2001.05140*.

Zhang, Zhen et al. (2019). 'Hierarchical Graph Pooling with Structure Learning'. In: *arXiv preprint arXiv:1911.05954*.

Zhang, Ziwei et al. (2018b). 'Timers: Error-bounded svd restart on dynamic networks'. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. 1.

Zhou, Bolei et al. (2016). 'Learning deep features for discriminative localization'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2921–2929.

Zhou, Yujing et al. (2019). 'Dynamic network embedding by semantic evolution'. In: *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, pp. 1–8.

Zhu, Linhong et al. (2016). 'Scalable temporal latent space inference for link prediction in dynamic social networks'. In: *IEEE Transactions on Knowledge and Data Engineering* 28.10, pp. 2765–2777.

Zou, Hui, Trevor Hastie and Robert Tibshirani (2006). 'Sparse principal component analysis'. In: *Journal of computational and graphical statistics* 15.2, pp. 265–286.