UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Structure characterization of complex networks for machine learning

Leandro Anghinoni

Tese de Doutorado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)

ICMC USP
SÃO CARLOS

Leandro Anghinoni

Structure characterization of complex networks for machine learning

Doctoral dissertation submitted to the Instituto de Ciências Matemáticas e de Computação – ICMC-USP, in partial fulfillment of the requirements for the degree of the Doctorate Program in Computer Science and Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Prof. Dr. Zhao Liang
Co-advisor: Prof. Dr. Israel Tojal da Silva

USP – São Carlos
July 2023

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

```
        Anghinoni, Leandro
A587s      Structure characterization of complex networks
        for machine learning / Leandro Anghinoni;
        orientador Zhao Liang; coorientador Israel Tojal da
        Silva. -- São Carlos, 2023.
           94 p.


           Tese (Doutorado - Programa de Pós-Graduação em
        Ciências de Computação e Matemática Computacional) --
         Instituto de Ciências Matemáticas e de Computação,
        Universidade de São Paulo, 2023.


           1. Machine Learning. 2. Complex Networks. 3.
        Community Structure. 4. Core-Periphery Network. 5.
        Graph Neural Network. I. Liang, Zhao, orient. II.
        Tojal da Silva, Israel, coorient. III. Título.
```

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Leandro Anghinoni

Caracterização da estrutura de redes complexas para aprendizado de máquinas

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientador: Prof. Dr. Zhao Liang
Coorientador: Prof. Dr. Israel Tojal da Silva

USP – São Carlos
Julho de 2023

ACKNOWLEDGEMENTS

I would like to thank my wife, for her unconditional support during these years. Also, I would like to thank my advisor, Prof. Liang Zhao, for his dedication to helping me in this work and his invaluable advises and insights.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001

RESUMO

ANGHINONI, L. **Caracterização da estrutura de redes complexas para aprendizado de máquinas**. 2023. 94 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2023.

Na última década, o aprendizado de máquina prosperou devido à avanços significativos na capacidade do hardware e no desenvolvimento de novos modelos. Modelos baseados em redes têm atraído bastante atenção recentemente por sua capacidade de aprender não somente com base nas características físicas dos dados (similaridade, distribuição, etc.) mas também com base no padrão de conexão entre os dados. Na busca de modelos melhores, a pesquisa evoluiu para incorporar a estrutura da rede no processo de aprendizagem. Alguns trabalhos recentes têm mostrado que explorar a estrutura da rede pode levar a melhores resultados de aprendizagem. Isto é feito capturando as conexões mais relevantes no processo de aprendizagem baseado na topologia da rede. Em vista disso, esta tese desenvolve quatro estudos para incorporar a estrutura da rede em algoritmos de aprendizado de máquina. No primeiro estudo, a estrutura da rede é utilizada para aprender padrões de séries temporais através de algoritmos de detecção de comunidades. O segundo estudo usa uma estrutura de rede *core-periphery* para representar dados onde uma das classes tem uma alta dispersão e é difícil de ser classificada por algoritmos tradicionais. Em outras palavras, introduzimos um método baseado em rede para representar o padrão de dados "sem padrão". O terceiro estudo propõe modelar um surto epidêmico através da predição de conexões em uma rede construída a partir de dados reais. Mostra-se que o isolamento social e o uso de máscaras pode diminiur o pico de casos de COVID-19. No último estudo, propomos um novo modelo de rede neural em grafo (*Graph Neural Network*) que combina a estrutura de comunidade dos dados do grafo e os vetores de características dos nós para gerar um *embedding* do grafo de forma rápida. A GNN proposta evita o problema de *over-smoothing* de métodos clássicos. Estes estudos mostram que a abordagem através de redes complexas pode superar várias deficiências de técnicas clássicas de aprendizado.

Palavras-chave: Aprendizado de Máquina, Redes Complexas, Estrutura de Comunidades, Redes Core-Periphery, Graph Neural Network.

ABSTRACT

ANGHINONI, L. **Structure characterization of complex networks for machine learning**. 2023. 94 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2023.

Over the last decade, machine learning has flourished due to significant advances in hardware capacity and model developments. Network based models have recently gained a lot of attention due to their capacity to learn not only from the physical features (similarity, distribution, etc.), but also from the connectivity pattern of the data. In the search of better models, the research has evolved to incorporate the structure of the network in the learning process. Some recent works have shown that exploiting the network structure can lead to better learning performance. This is done by capturing the more relevant connections in the training process based on the network topology. In light of this, this thesis carries out four studies to incorporate the network structure in machine learning algorithms. In the first study, the network structure is used to learn time series patterns via community detection algorithms. The second study uses a core-periphery network structure to represent data where the data within one of the classes has a very high dispersion and is hard to be classified by traditional algorithms. In other words, we introduce a network-based method to represent data pattern of the data "without pattern". The third study aims to model an epidemic outbreak via link prediction in a network constructed from real data. We find that social isolation and wearing masks can effectively decrease the COVID-19 epidemics peak. In the final study, we propose a novel Graph Neural Network (GNN) model by combining the community structure of the underlying data graph and the feature vectors of the nodes to generate a graph embedding in a fast way. The proposed GNN can avoid the over-smoothing drawback of classic ones. These studies show that complex network approach can overcome various shortcomings of classic learning techniques.

Keywords: Machine Learning, Complex Networks, Community Structure, Core-Periphery Network, Graph Neural Network.

LIST OF FIGURES

CONTENTS

CHAPTER 1

INTRODUCTION

Since the development of the first computing hardware, human have wondered if a machine would ever learn like humans do. As a consequence, Machine Learning (ML) has become one of the greatest endeavors in the scientific field. ML concerns proposing methods to enable a machine learn from past experience, in a similar way as humans do. This includes, for example, inferring the class of a new instance after reasoning over a set of classified instances (MITCHELL *et al.*, 2007). In a higher level, a lot o similarities can be observed between the way that a human being learns and the way that the state-of-art ML algorithms work. However, modeling such kind of tools is still a challenge since the human decision process cannot always be converted to machine language. Still, ML algorithms are becoming more present in our daily life in a very fast pace and in a variety of areas, such as shopping, construction, medical, finance and many others.

In a general way, a ML algorithm aims to learn a target function from a set of training examples based on a cost function that reduces the error to the lowest level in the training set (BISHOP; NASRABADI, 2006; LECUN; BENGIO; HINTON, 2015; ZHOU, 2021). Although many different categorizations have been proposed over the years, these algorithms can be divided into four big groups, depending on how much is known about the training samples: (i) Supervised, (ii) Semi-supervised, (iii) Unsupervised and (iv) Reinforcement (SARKER, 2021). In the first group (supervised) all the training samples are labeled. The learning objective is to construct a classifier to predict the labels of new data samples. In these algorithms, usually the larger the data-set, the better the label prediction performance. The downside is that the computational cost tends to be high with large training sets. In the second group (semi-supervised), a small portion of the training set is labeled and the labels are propagated to the data samples without labels. Such algorithms are quite useful for problems with big data-sets, which would require a large amount of manual annotations. In the unsupervised algorithms, no data sample is labeled. In this case, the learning process tries to find out intrinsic data patterns, such as clusters. Finally, the reinforcement algorithms work by applying a reward and penalty rule to evaluate the optimal behavior in a given context.

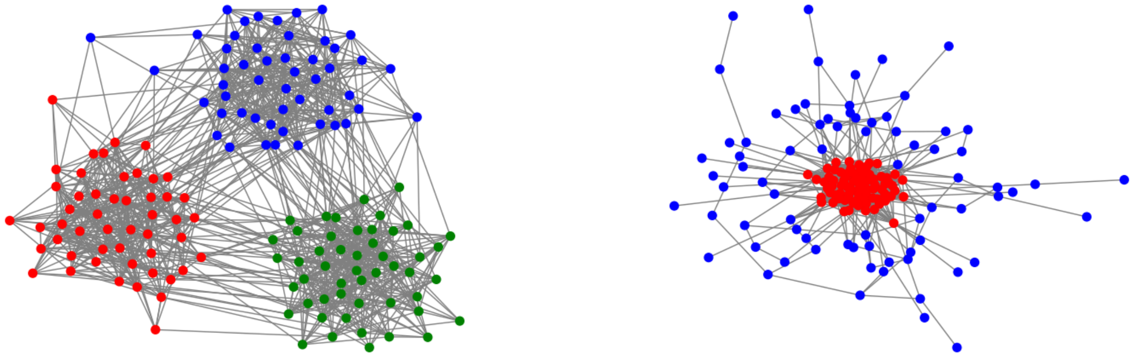
The majority of the classic and the state-of-the-art ML algorithms works in a similar way when grouping the data samples into different classes or clusters: Splitting the data space or the feature space into sub-spaces that best separates each class or cluster (BISHOP; NASRABADI, 2006; LECUN; BENGIO; HINTON, 2015; ZHOU, 2021). This approach has proved to be a very robust strategy, specially when the training data is vast. However, such a strategy can suffer from the large variations of data patterns and the high complexity of class (cluster) boundaries. Also, these algorithms can have high computational complexity (LECUN; BENGIO; HINTON, 2015) in high dimensional data or feature spaces. Moreover, classic ML techniques have difficulty to interpret the learning results, since they lack a mechanism to characterize the large region or global data patterns.

In the recent years, there is an increasing interest in the development of new ML algorithms which are able to capture not only the physical features, but also the semantic relationship of the data. Such an approach can learn data pattern with complex geometrical forms (SILVA; ZHAO, 2012a; SILVA; ZHAO, 2015; COLLIRI *et al.*, 2018; CARNEIRO; ZHAO, 2017). A powerful way to capture various kinds of relationship in the data is through the topology structure of a complex network.

A complex network is a large scale graph with non-trivial connections [1] (STROGATZ, 2001; BOCCALETTI *et al.*, 2006). They are in the heart of the complex systems for their interdisciplinary, quantitative, mathematical and computational nature. Complex networks play an important role in real world applications due to their ability to capture the underlying relation between entities (BARABÁSI, 2013). The mesoscale structure of a complex network can be very useful to reveal data patterns, which are hard to be uncovered in regular space, such as the Euclidean space.

A complex network structure can refer to several types of arrangements. One of the salient features of complex networks is the presence of communities. The notion of community in networks is defined as a sub-graph whose nodes are densely connected within itself, but sparsely connected with the rest of the network. Community detection in complex networks has turned out to be an important topic in graph mining and data mining (FORTUNATO, 2010; NEWMAN; GIRVAN, 2004; DANON *et al.*, 2005). A lot of efforts has been spent to develop efficient community detection methods. For a comprehensive review of this topic, see (FORTUNATO, 2010). A network with a defined community structure presents regions with denser connections, which can be measured by a modularity index (NEWMAN; GIRVAN, 2004). In the communities, the intra-community degree of every node is much higher than the inter-community degree. Another interesting feature of complex networks is the core-periphery structure (BORGATTI; EVERETT, 2000). A core-periphery structure, on the other hand, supposes a higher density

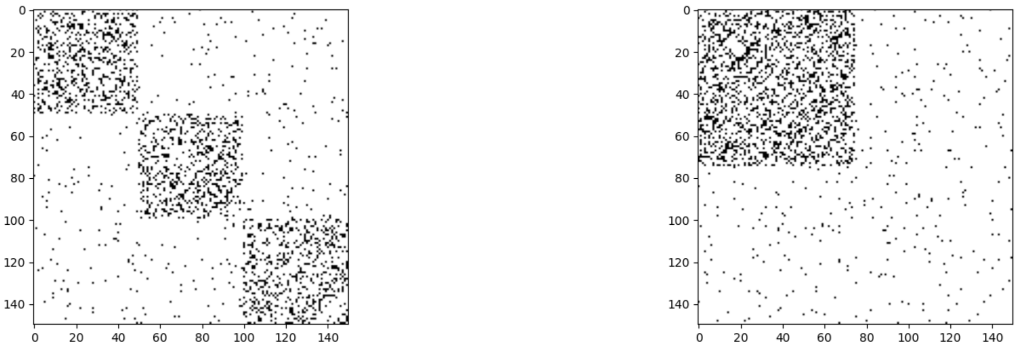
[1] Complex network and graph share the same definition. Therefore, the two terms are interchangeable in this document



(a) Network with a community structure. Each color represents a different community, with a higher intra-class degree and lower inter-class degree.

(b) Network with a core-periphery structure. The red samples represent the core and the blue samples the periphery.

Figure 1 – Examples of two different network structures.



(a) Adjacency matrix of a community structured network.

(b) Adjacency matrix of a core-periphery network.

Figure 2 – Examples of adjacency matrices for two different types of network structure. A black dot represents a connection between two nodes. The relation in data is captured by the interconnections of the whole system.

within the core classes and a low density in the other regions, i.e., the intra-class degree of the periphery class is also low. These two kinds of complex network structures are illustrated in Figure 1 and their respective adjacency matrix in Figure 2.

In this thesis, we will explore the community and the core-periphery structures to represent data patterns for various ML tasks.

In the domain of ML, the topological network structure is very useful to detect various forms of clusters or classes by an agglomeration or classification algorithm. As a consequence, network-based methods in learning tasks have become a very active area of research with a variety of applications, such as supervised learning (IOSIFIDIS; TEFAS; PITAS, 2015; NICKEL *et al.*, 2015; ZHANG; CUI; ZHU, 2020), semi-supervised learning (CHAPELLE; SCHOLKOPF;

ZIEN, 2009; GONG *et al.*, 2015; SILVA; ZHAO, 2012a; ZHANG *et al.*, 2014; ZHU, 2005), data clustering (CHEN; LV; YI, 2017; FORTUNATO, 2010; SILVA; ZHAO, 2012b; SILVA; ZHAO, 2012a; VERRI; URIO; ZHAO, 2016), graphs and sub-graphs matching (ZHANG *et al.*, 2015), regression (NI; YAN; KASSIM, 2010), feature selection (BUNKE; RIESEN, 2011), dimensionality reduction (RIESEN; BUNKE, 2009), interpretation of ML via visualization (ZHANG *et al.*, 2018). The emergence of machine learning based on complex networks is explained by the inherent advantages that the representation of data as networks provide, allowing to capture spatial, topological, dynamic and functional relationships of large data sets. Some encouraging results have already been obtained by the research group led by the advisor of this thesis in this direction (SILVA; ZHAO, 2016; BREVE *et al.*, 2011). For example, a particle competition model for community detection, data grouping and semi-supervised classification was developed (SILVA; ZHAO, 2012b; SILVA; ZHAO, 2012a; VERRI; URIO; ZHAO, 2016). The basic idea of the particle competition model is as follows: We put some particles in a given graph, then, these particles travel in the graph and try to dominate as many nodes as possible. At the same time, each particle competes with other particles to avoid intrusion to its territory. At the end, each particle is expected to occupy a sub-graph corresponding to a community. The model is inspired by real systems, such as election champions, competition for food, water or territory among animals, etc. The great advantage of the proposed technique comes from its ability to identify arbitrary form and distributed data groups. Another salient feature of the model is the walking behavior of the particles. At each step, each particle chooses a neighbor node to visit using a combined random walking and preferential walking rule. Random walk means that a particle randomly selects a neighbor to visit. It represents an exploratory behavior, i.e., a particle tries to discover new territory, while preferential walk means that a particle prefers to visit a node already dominated by itself. It characterizes a defensive behavior, i.e., the particle would like to strengthen its own base instead of exploring the whole network. The particle competition model presents a robust while efficient way to identify the data graph structure, specifically the modular structure. In the case of supervised learning, a high-level data classification technique was proposed in (SILVA; ZHAO, 2012a) and extended in (CARNEIRO; ZHAO, 2017; CARNEIRO *et al.*, 2019; SILVA; ZHAO, 2015). In this scheme, the low-level classification can be implemented by any traditional classification technique, while the high-level technique explores the complex topological properties of the network built from the input data. A salient feature of high-level classification is the classification of data based on the pattern formation of the input data instead of considering only the physical attributes, and is therefore referred to as a high-level classification.

Deep learning has revolutionized many machine learning tasks in recent years, such as object detection (REDMON *et al.*, 2016; REN *et al.*, 2015), machine translation (LUONG; PHAM; MANNING, 2015; WU *et al.*, 2016), and speech recognition (HINTON *et al.*, 2012). The data sets used in deep learning are typically represented in the Euclidean space. However, there is an increasing number of applications, where data samples are generated from non-Euclidean domains and are represented as graphs with complex relationships between objects. For example, in e-commerce, a graph-based learning system can exploit the interactions between users and products to make highly accurate recommendations. In chemistry, molecules are modeled as graphs, and their bio-activity needs to be identified for drug discovery. In social networks, people are linked to each other and, at the same time, they can be categorized into different groups. The complexity of graph data has imposed significant challenges on the existing machine learning algorithms. This is because some important operations (e.g., convolutions) are easy to compute in the structured domain, for example, in images, but difficult to apply to the graph domain. Based on Convolutional Neural Networks (CNNs) and graph embedding, variants of Graph Neural Networks (GNNs) are proposed to collectively aggregate information from graph structure. Thus they can model input and output consisting of elements and their dependency (see (WARD *et al.*, 2022; ZHANG; CUI; ZHU, 2020; WU *et al.*, 2020; ZHOU *et al.*, 2020) and references therein). The main component of GNNs is the graph embedding generation to capture the relationship among nodes. One of the important mechanisms for generating graph embedding is by means of message passing, i.e., propagating feature vectors of the nodes to a certain range of neighbors in the graph. However, current message passing mechanisms are indiscriminate and propagate signals locally, resulting in a shallow neural network mostly with three layers, which limits us from taking the advantages of deep learning. Moreover, current message passing mechanisms also lead to over-smoothing phenomenon, resulting in gradient vanishing. Therefore, still in this thesis, we will develop a new GNN by applying particle competition mechanism for graph embedding generation to overcome the above mentioned shortcomings of GNNs.

The studies developed in this work explore novel ways to incorporate the network structure information into machine learning models. The first paper has been developed in the first year of my doctorate. In this paper, we explore the topological structures to represent time series patterns, specifically, we use hierarchical community structure of a complex network constructed from time series data. In the second year of my doctorate, Covid-19 became an urgent topic of research and at that time, we modeled the viral spread using the SIR model in a networked environment. We conceived the model together and my contribution on every step was focused on the network construction and modeling. Later, while studying some medical data, we realized some problems may present a highly dispersed class, i.e., the instances do not share any similarity. To deal with this, we have explored core-periphery networks to represent such problems. This time, I teamed up with other researchers to write a paper on x-ray image classification. Finally, in the last years of my doctorate, we have dedicated ourselves to study Graph Neural Networks (GNNs) and more advanced medical image processing problems, such as whole slide image classification in image pathology. We realized that network science can make a good contribution to GNN modeling, specifically, the message passing in a structured graph could improve the GNN$'$ s ability to learn global and local features. Therefore, we have proposed a way to combine the information generated in the clustering process with the original features of the data.

In the next Section I detail the objectives and motivations of all the works presented here.

1.1 Objectives

The main goal of this work is to characterizing data patterns using various complex network structures and metrics in the topological space. The underlying hypothesis of all the papers presented in this document is that learning the topological structure of the data can lead to a better understanding of the relationship among data samples, consequently, can overcome some shortcomings of the current ML techniques.

More specifically, I tackle four specific topics in the following Sections:

- Identification of time series patterns, by converting stochastic time series to state transition networks and clustering it hierarchically, for trend prediction. In other words, we transform the time series from the time-frequency space to the topological space and we use the community structure to represent each time series pattern.

- Predicting a viral disease spread by fitting a network to real data and estimating the spread using an early-time dynamic of the SIR model to simulate the spread. The network structure is changed to test the efficacy of public health measures. The paper shows that social isolation and wearing masks can effectively decrease the COVID-19 epidemics peak, which has special importance at that confusing time. For this reason, this article generated big repercussion in the main Brazilian national medias, such as Agência FAPESP, UOL, Revista Galileu, G1, Folha de São Paulo, and EPTV.

- Characterization of data patterns with high dispersion through core-periphery network structure. The hypothesis here is that the core structure can capture the pattern of normal samples and the periphery the pattern of abnormal samples, that present high dispersion. In this way, we propose a method to characterize data patterns for those kinds of data "with out patterns";

- Improvement of graph embeddings by learning the graph structure prior to performing the message passing of a graph neural network. Due to its high robustness and high efficiency (linear time complexity), we hope the new GNN to be developed contains the following features: 1) The generated graph embedding captures not only local relationship between nodes but also global structure of the graph. This feature is useful to classify data samples distributed in the border or highly mixed region. 2) More iterations that the particles walk in the underlying graph correspond to more layers of the neural network. In this way, we will get a real deep GNN, consequently, improving its learning performance. 3) The high efficiency of the Particle Competition model allows the new GNN to process large-scale data graphs.

1.2 Motivations

The motivations behind each of the works presented here are presented in the following subsections.

1.2.1 Time series pattern identification

Identifying time series patterns is a vital task for many scientific and applied fields. The correct assessment of the data is crucial to develop models that not only can identify the patterns but also predict the upcoming ones. Traditional statistical models are very effective when the data can be decomposed in trends, cycles and noise (RANI *et al.*, 2014; BAHETI; TOSHNIWAL, 2014). Unfortunately, many important real world applications are presented in the form of stochastic-like data, i.e., the data presents stochastic characteristics but, eventually, a sequence of patterns can be captured by modern machine learning algorithms, that track temporal evolution of the data. Although neural network models (LSTM) have been quite successful with a variety of applications, when it comes to more complicated data, such as stock market prices, modeling the patterns and their temporal relation is still a challenging task (SIAMI-NAMINI; TAVAKOLI; NAMIN, 2018).

1.2.2 Prediction of viral disease spread

Predicting the spread of a new disease can be a challenging task, specially in the first moments of an outbreak. Traditional models, such as the SIR model, should not be applied without considering the spatial distribution of the cases (KEELING; EAMES, 2005; STEGEHUIS; HOFSTAD; LEEUWAARDEN, 2016). Early stage behavior is also hard to model (LIU *et al.*, 2023) and the effect of public health measures are also difficult to incorporate in traditional model due to a lag in its outcome.

1.2.3 Classification of data with high dispersion

In the supervised learning paradigm, the algorithm learns a function from the labeled samples that maps the data to the classes, which is later used to classify unlabeled data. Several models have been developed, including a number of state-of-the-art algorithms (BISHOP; NASRABADI, 2006; LECUN; BENGIO; HINTON, 2015). However, they all work in a similar way, by dividing the data space into sub-spaces that best divide the training data. In this scheme, strong distortions in the boundaries lead to poor performance, as well as high dispersion in some classes, or the presence of classes without a defined pattern. Moreover, the semantic relationships are not considered and the advanced models tend to be hard to interpret (ZHANG; ZHU, 2018).

1.2.4 Capturing global information for graph neural network learning

Graph Neural Networks (GNNs) are a recent type of neural network that considers the relationship contained in the data (WU *et al.*, 2020). In a general GNN architecture, the information is passed from one node to its neighbors through a message passing function and an aggregation function is used to update the node‘s embedding. The model then learns an inductive function based on the error propagated by each epoch, as in a neural network. GNNs are inherently shallow. This means that they are very good at learning local information, but fail too learn global information, as this can lead to the so-called over-smoothing (every node ends up with a similar feature) (OONO; SUZUKI, 2019).

1.3 Outline

In the next chapters, four papers are presented. All of them are studies based on the structure characterization of complex networks to address the objective and motivations listed in the previous sections.

In Chapter 2 a method to identify time series patterns based on complex networks communities is proposed. The paper also shows that the network partitioning process can be done progressively in order to capture local and global patterns. As a result, the model is able to identify long and short trend in an artificial data-set and correctly classify up and down trend in a stock price chart.

In chapter 3, a method is proposed to model the Covid outbreak in Brazil. The model constructs a transmission network by fitting the network links to real data of the cities in the early stage of the outbreak. Then an early-dynamic SIR model is applied on top of the constructed network to simulate the evolution in every city of the network. The effectiveness of public measures is evaluated by inspecting the change in the Covid-19 epidemic peak, when the network structure is changed to reflect the omission of this public measures.

Next, in chapter 4, contributions on classification of high dispersion data is shown. A method is proposed based on the structure of a core-periphery network. Unlike traditional methods, the proposed framework uses high-level classification to group the normal data in the core structure and the high dispersion data in the periphery structure. The model is then applied to a data-set of x-ray chest images containing normal images and Covid images (high dispersion class) and is able to correctly classify new unlabeled instances.

In chapter 5, a method is proposed to capture the structure of the network and embed this information in the feature of each node of the network. To accomplish that, it is proposed a pre-processing step, based on the particle competition and cooperation algorithm, that assigns the most probable cluster to every node. This information is included in the node feature, so that the GNN starts the learning process considering this global feature.

Finally, in chapter 6, the final remarks of this work are presented. The works presented here are shortly evaluated and possible future works are outlined.

CHAPTER

2

TIME SERIES PATTERN IDENTIFICATION BY HIERARCHICAL COMMUNITY DETECTION

Author contribution statement

L.A. and D.A.V.O conceived the method and conducted the experiments. T.C.S and L.Z supervised the project. All authors discussed the ideas and the results. All authors reviewed the manuscript.

Eur. Phys. J. Spec. Top.
https://doi.org/10.1140/epjs/s11734-021-00163-4

THE EUROPEAN PHYSICAL JOURNAL SPECIAL TOPICS

Regular Article

Time series pattern identification by hierarchical community detection

Leandro Anghinoni[1,a], Didier A. Vega-Oliveros[2,3], Thiago Christiano Silva[4,5], and Liang Zhao[4]

[1] ICMC, University of Sao Paulo, São Carlos, SP, Brazil
[2] Institute of Computing, University of Campinas, Campinas, SP, Brazil
[3] Center for Complex Networks and Systems Research, Luddy School of Informatics, Computing and Engineering, Indiana University, Bloomington, IN, USA
[4] FFCLRP, University of Sao Paulo, Ribeirão Preto, SP, Brazil
[5] Universidade Católica de Brasília, Brasília, Brazil

Received 29 November 2020 / Accepted 21 April 2021

Abstract Identifying time series patterns is of great importance for many real-world problems in a variety of scientific fields. Here, we present a method to identify time series patterns in multiscale levels based on the hierarchical community representation in a complex network. The construction method transforms the time series into a network according to its segments' correlation. The constructed network's quality is evaluated in terms of the largest correlation threshold that reaches the largest main component's size. The presence of repeated hierarchical patterns is then captured through network metrics, such as the modularity along the community detection process. We show the benefits of the proposed method by testing in one artificial dataset and two real-world time series applications. The results indicate that the method can successfully identify the original data's hierarchical (micro and macro) characteristics.

1 Introduction

Time series pattern recognition is a broad field of research. It has advanced over the last decade with the developments of techniques tackling the time series in different domains [1,2]. An important area of research is analyzing the time series with the aid of a complex network [3–13]. Studies on network topology have remarkably advanced our understanding of this domain, both in terms of data complexity—that went from a single univariate time series to complex data flows containing concept drifts [3–6]—and in terms of proposed frameworks [13–17,28,29]. Most of these frameworks can be generalized by a two-step process of converting the data into a network (mapping) and analyzing it through network metrics.

Once the data are mapped to a network, it can be analyzed through network metrics. In particular, detecting the community structure of the network has gained a lot of attention since some works have shown that the community structure can represent the data patterns and reflect changes along time [17–20]. More recently, some works have shown that, intuitively, the communities can contain information about structural patterns of the original data [5,13], even exploring repetition cycles for stochastic times series [29]. Although previous works have set well-established tools for modeling and mining time series based on networks, there is a lack of understanding of the relationship between the discovered patterns and the original data, such as their hierarchy and recurrent cycles, which is still an open and challenging task. Moreover, in many cases, the results depend on how well tailored is the rule for mapping the data according to the problem.

Here, we propose a more general method to identify time series patterns in multi-scale levels based on the hierarchical community representation in a complex network. The construction process follows the general rule of adopting the largest correlation threshold that leads to a less fragmented network. The presence of repeated hierarchical patterns is then captured through network metrics concerning the modularity along the community detection process. This way, we can evaluate how the modularity impacts the resulting clustered network. Moreover, the method allows visualizing the relationship between the discovered multi-scale patterns and the original time series data, according to the detected communities.

In summary, the main contributions of this work are threefold:

1. We explore how the time series's characteristics are carried to the network structure by detailing the parameters setting of the proposed framework.
2. We detail the community formation process using a dendrogram, making explicit the hierarchical rela-

[a] e-mail: anghinoni@usp.br (corresponding author)

Published online: 03 June 2021

tion of the time series patterns and its multi-scale properties.

3. We map the communities back to the time series to show the patterns' temporal relation.

The remainder of this paper is structured as follows: Sect. 2 presents a brief description of the methods employed in this work to analyze a given time series through its community structure. In Sect. 3, we present our proposed method is presented and its details on each of the steps. Section 4, we have the results in an artificial dataset to illustrate some of the main characteristics of our approach. Section 5 presents the results of applying the method in two real data-sets. Finally, conclusions are presented in Sect. 6.

2 Materials and methods

2.1 Time series analysis through network topology

To study a time series through network topology, the data have to be converted into a network's representation and then analyzed over the different aspects present in this type of structure, such as the centrality measures [10,14], minimum paths [4], network's modularity [13,29], among many others. Mapping a time series into a complex network can be done by several different methods, comprehensively described in [20]. The three main approaches, however, consists of (i) cycle networks, (ii) visibility graphs, and (iii) recurrence networks.

In a cycle network [21], the time series is divided into several disjoint parts. These parts become the network nodes, which are then connected based on some rule, like the phase space distance or the Pearson correlation between each segment.

The visibility graph [15] considers the time series as a landscape. In this approach, each point of the time series becomes a node, which is then connected if they can 'see' each other with no interference of any other point. Many variations of the original idea have been proposed by creating specific rules for considering the time series's points and how they connect (some methods allow that two points are not completely visible, i.e., there can be a certain amount of obstruction between them).

Finally, the recurrence network [16] considers the time series as a sequence of phase space vectors. This method has flourished over the last years due to the possibility of adaptation to different problems since the state vector can assume any proposed shape.

These three approaches have inspired many different works and led to several variants of the original ideas. Besides, many applications can be found in the literature, in a wide variety of research fields, such as finance (stock time series), medical (EEG/fMRI signals time series), engineering (fluid flow time series), meteorology (temperature time series), and others [20].

2.2 Community detection

Community detection methods have also developed over the last decade to adapt to various problems [22]. These methods can be divided into two big groups: (i) divisive (or top-down) algorithms and (ii) agglomerative (or bottom-up) algorithms. In the divisive algorithms, the whole network is considered to be one community. At each iteration, the algorithm detaches a new group of nodes until the network modularity is maximized or the number of desired communities is reached. Several algorithms follow this idea, such as in [23]. On the other hand, the agglomerative algorithms consider each node to be a community at the beginning of the process. At each step, a specific number of nodes are clustered. The clustering process stops either when the maximum modularity is reached [24], when the number of communities is reached or when the system reached equilibrium [5]. The algorithm can also take into consideration the overlapping nature of the communities [22,25,27] or consider that a node can only belong to one community.

One way to measure the quality of a graph partition is to calculate its modularity Q. The modularity measures the difference between the connections observed in a given network and the expected connections observed in a random network [26]. The idea stems from the fact that the probability of connection between two nodes of the same community is higher than for nodes in distinct communities. Formally, the modularity is defined as follows:

$$Q = \frac{1}{2m} \sum_{vw} \left[A_{vw} - \frac{k_v k_w}{2m} \right] \delta\left(c_v, c_w\right) \tag{1}$$

$$\delta\left(c_v, c_w\right) = \begin{cases} 1, \text{ if } v \text{ and } w \text{ belong to the same community} \\ 0, \text{ otherwise} \end{cases} \tag{2}$$

where m is the total number of edges, A_{vw} is the actual number of edges between v and w, k_v and k_w are the degrees of nodes v and w. Hence, the sum term of the equation calculates the difference from the actual edges and the expected number of edges in a random distribution over every pair of nodes and weights them considering the total number of edges and whether they are in the same community or not. The higher the value of Q, the less random are the connections, i.e., the network presents structured communities. We use the modularity as a reference indicator of the network's quality in this work, as described in the next Sect. 3.

Probabilistic approaches can also be applied to this task since the size of real networks can impose a computational limitation on deterministic methods. In [27], for example, a Bayesian model is presented to tackle massive networks by computing the community of parts of the full graph and updating the communities' estimate based on the previous step.

3 Proposed method

Here, we propose a framework to understand how the community structure reflects the original data's patterns. A good community structure is found in a high modularity network, meaning that the constructed network presents distinguishable patterns (represented by each community). Therefore, every parameter used to construct the network has to be set to maximize the network modularity.

Having that in mind, the proposed framework will rely on traditional methods to keep the number of parameters low (two to be more precise). Besides, we analyzed the communities, including their relations and hierarchy, by making gradual decomposition of the communities into smaller ones. The method is summarized in the flowchart of Fig. 1.

3.1 Network construction

First, we map our data using a sliding window to determine the Pearson correlation between every data point, similarly to the method proposed in [28]. Therefore, let w be the size of the sliding window, the network will be formed by $N - w$ nodes, where N is the size of the time series. At first, every node is connected, and the weight of the edge is equal to the Pearson correlation ρ between the two data points.

Given a time series $S(t), t = 0, 1, 2, \ldots, N$, we define the segments $S_i = \{S\big(i - \lfloor w/2 \rfloor\big), \ldots, S\big(i + \lfloor w/2 \rfloor\big)\}$ and S_j in a similar way, with $i, j = \{x \in \mathbb{Z} \mid \lfloor w/2 \rfloor \leq x \leq N - \lfloor w/2 \rfloor\}$ and $i \neq j$. Then, the edge e_{ij} is the similarity between S_i and S_j, which is calculated by the Pearson correlation as:

$$\rho(S_i, S_j) = \frac{\mathbb{E}[S_i S_j] - \mathbb{E}[S_i] \cdot \mathbb{E}[S_j]}{\sqrt{\mathbb{E}[{S_i}^2] - (\mathbb{E}[S_i])^2} \cdot \sqrt{\mathbb{E}[{S_j}^2] - (\mathbb{E}[S_j])^2}} \tag{3}$$

However, in the method, weak connections are not considered since they create noise in the constructed network. Therefore, a threshold $\rho_{\min}$ is used in the following manner:

$$e_{ij} = \begin{cases} 1, & \text{if} \quad \rho \geq \rho_{\min} \\ 0, & \text{if} \quad \rho < \rho_{\min} \end{cases} \tag{4}$$

where e_{ij} is the edge between nodes i and j.

It is important to notice that setting a high $\rho_{\min}$ can generate a network with more than one component and even disconnected nodes. Therefore, we propose the following rules to set the parameters w and ρ.

3.1.1 Setting the parameters

First, we set the sliding window value w used to generate the segments to be compared. By definition, a time series cycle can be defined as the data between two peaks. Also, the segment between two peaks should contain two main patterns, a decreasing one and an increasing one. Therefore, let k be the number of peaks in the time series, we set w to:

$$w = \left\lfloor \frac{N}{2k} \right\rfloor \tag{5}$$

Setting the Pearson correlation $\rho_{\min}$ requires evaluating the constructed network. The value will depend on the original data's characteristics, like the amount of noise and the presence of repeated patterns. High $\rho_{\min}$ will be used in time series with little noise and repeating patterns, whereas stochastic time series will require lower values for $\rho_{\min}$. A direct way to set $\rho_{\min}$ is by setting it to 1 and gradually decrease the value until the constructed network is composed of one component. At this value of $\rho_{\min}$ every segment is connected to at least another segment and can be assigned a community in the clustering process.

3.2 Analyzing community patterns

To study the community patterns present in a given data-set, we perform a progressive partitioning using the Newman–Girvan [23] community detection method. This means that instead of stopping the process at the higher modularity, we measure the modularity at each step until the network is decomposed into communities formed by single nodes. By doing so, it is possible to understand the following properties of the data:

- How significant the sub-patterns are;
- How the modularity behaves in terms of community number, instead of just looking at the maximum modularity (sometimes, in real applications, fewer communities can yield more stable patterns without compromising in terms of modularity);
- How the communities (or patterns) are linked hierarchically. A pattern can be composed of several sub-patterns in a time series, and the community splitting procedure should generate a dendrogram representing that.

In the next section, these characteristics will be studied, where we use an artificial data-set to verify them.

4 Experiments

Before applying the framework to real data-sets, some experiments were conducted on an artificial time series to evaluate the properties mentioned in the previous section.

To test the proposed method on a controlled data-set, we introduce the artificial time series S:

$$S(t) = sin(t) + 2sin(\frac{t}{10}), 0 < t < 200 \tag{6}$$

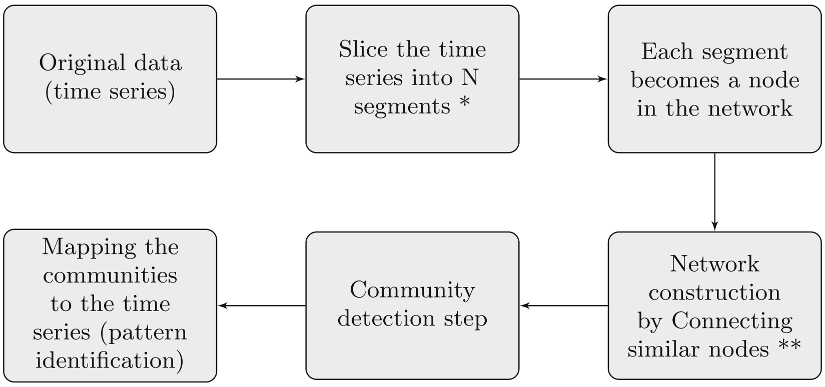

Original data (time series)
Slice the time series into N segments *
Each segment becomes a node in the network
Network construction by Connecting similar nodes **
Community detection step
Mapping the communities to the time series (pattern identification)

Fig. 1 Flowchart of the proposed framework. *The number of segments depends on the size of the window used in the experiment. **The level of similarity is also a parameter to be set. Both of these parameters will be detailed further in this paper

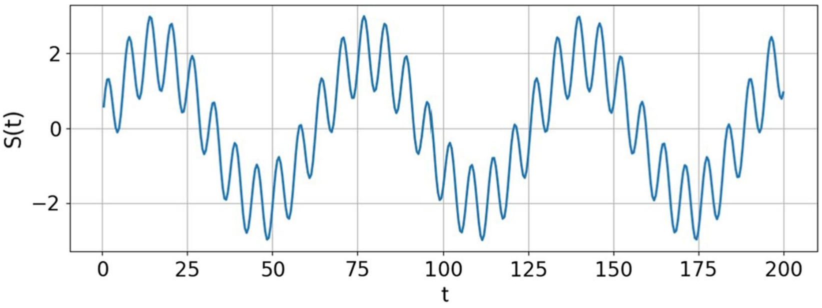


Fig. 2 Artificial dataset used to depict each step of the proposed method

where t can be set to any interval, we used the interval between 0 and 200, with a step of 0.5 between each data point. The data-set can be seen on Fig. 2.

To construct the network, we need to set two parameters, w and $\rho_{\min}$. The data contain three full cycles and 400 time periods, since S is calculated with a step of 0.5. Therefore, by using the rule described before, $w = \lfloor 400/(2 \cdot 3) \rfloor = 66$. Once w is set, we study the best value for $\rho_{\min}$. In Fig. 3a, we can see the size of the main component c_s as a percentage of the total number of nodes for the window that we set before. In this case, $\rho_{\min}$ can be set to 0.96, a high value, since the artificial data has no noise and clear repeating patterns.

4.1 Analyzing community patterns

In the next experiment, we performed the communities' gradual decomposition into smaller ones until the modularity decreases below the peak value. In Fig. 3b, we can see that, for $S(t)$, the highest modularity is reached when the network is divided into 19 communities. However, it is important to notice that the modularity increases rapidly from 2 to 10 communities. This indicates that breaking the patterns in more than 10 communities might add little information to understand the time series (i.e., the artificial one). For these experiments, $w = 66$ and $\rho_{\min} = 0.96$, as discussed before.

The next step is to analyze how the communities are formed from the start of the divisive process until it reaches the peak modularity (19 communities). Figure 4 shows a dendrogram of this process, depicting the top-down process of how the communities are formed.

To depict how this method classifies the time series into different patterns, depending on the number of communities, we plotted the time series corresponding to two different cuts, two and nineteen communities. Figure 5 shows the classified time series. Notice that when the network is divided into two communities, the time series classification reflects only the two major patterns (up trend in green and down trend in blue in Fig. 5a). All the smaller patterns inside the major pattern are grouped (which was also verified by the dendrogram). When we continue the partitioning process (until we reach 19 communities), sub-patterns are detected. In the Fig. 5b, we can see that the major up and downtrends were divided into sub-patterns representing the smaller fluctuations. For the sake of visualization, we have plotted only 6 of the 19 communities. The major up trend (green in Fig. 5a), for example, is composed of two sub-patterns, green for the short up trend and magenta for the short down trend. The major down trend is also composed of two sub-patterns, which are represented by different communities, in this case depicted in yellow and red, respectively. Also, we can see patterns indicating the reversal of the major trends, such as the blue that shows the end of a long up trend and and the purple that show the end of a long down trend. This picture also shows the temporal relation of the communities and how they repeat over time, since the patterns always appear in the same position in relation to the longer cycle.

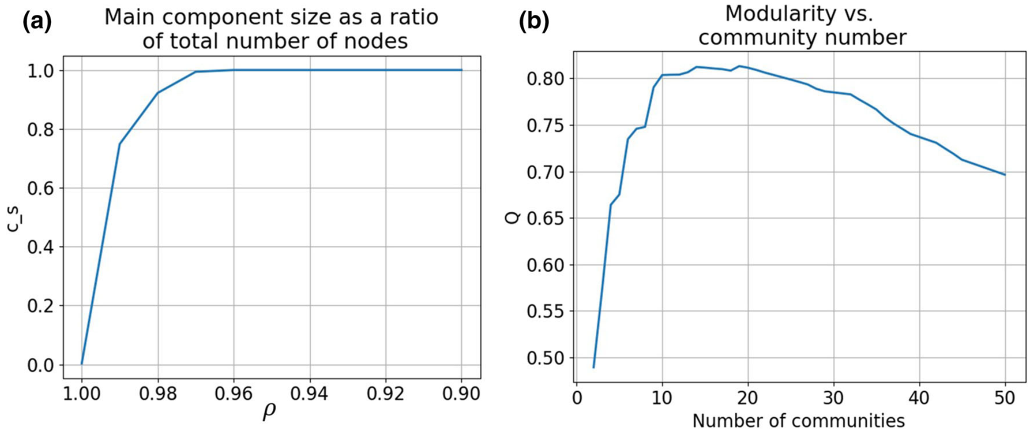
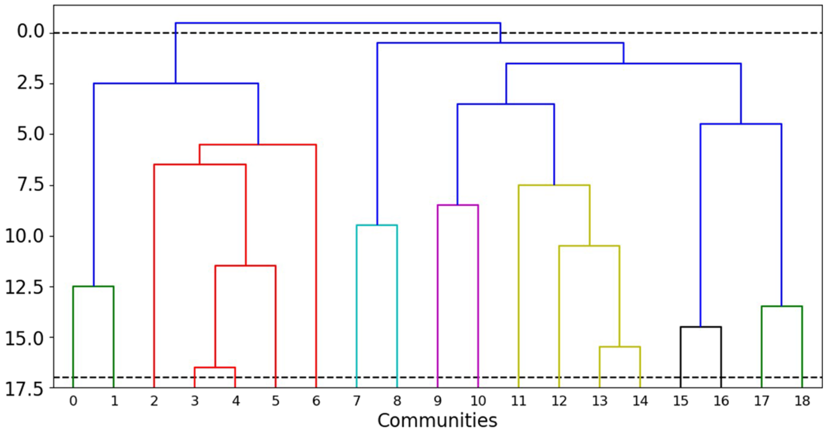


Fig. 3 Parameter used to set w and ρ for the artificial dataset. **a** Size of the main component c_s as a percentage of the total number of nodes; **b** modularity variation along the community detection process

Fig. 4 Dendrogram of the progressive partitioning process. From top to bottom, the network is divided until it reaches the peak modularity at 19 communities. We can see the hierarchy between each community in the dendrogram and by following the order in which the communities are split, we can infer the connection strength between them



5 Application

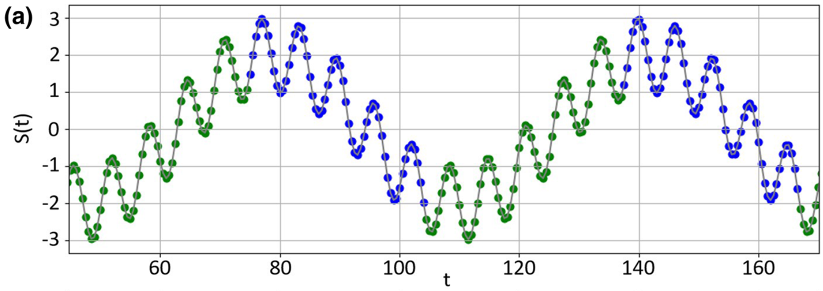
In the previous section, we tested the proposed method in an artificial time series $S(t)$ and analyzed it thoroughly using the community structure of the constructed network. In this section, we apply the same framework to real data-sets. The choice of w and $\rho_{\min}$ were made following the ideas presented in the previous section.

First, we applied the method to a temperature dataset [30]. This time series presents a cyclic behavior but contains a lot of noise, unlike the artificial data-set studied before. A period of 1000 days was analyzed—roughly 3 years. The parameters w and $\rho_{\min}$ were set to 166 and 0.85 respectively. Figure 6a shows the parameter setting charts. As we can see, the noise plays an important role in the value of $\rho_{\min}$, requiring the value to be much lower to generate a single component network.

In Fig. 6b, we can see that the best division is obtained with 13 communities. However, in this case, the modularity increase obtained by dividing the dataset into more than two patterns is marginal. This indicates that the data contains two important patterns, and the other 11 are not very relevant. To visualize this, we mapped back the communities to the time series in Fig. 7. As we can see, the other eleven patterns are reversal patterns. They indicate the end and the beginning of the two main patterns and that the reversal happens in different ways, given that the pattern is not always the same.

Next, we used stock market data, specifically a period of the Bovespa index (main Brazilian stock market index). Like any stock market time series, this time series is regarded as being generated by a stochastic process, which should impose some challenges to the proposed framework. This data-set was further explored in a previous work [29].

A period of 500 observations was used, and the parameter selection is based on Fig. 8a. The parameters w and $\rho_{\min}$ were set to 20 and 0.70, respectively. In this case, the $\rho_{\min}$ is even lower due to the nature of the data. The best division is obtained with 18 communities as indicated in Fig. 8b.



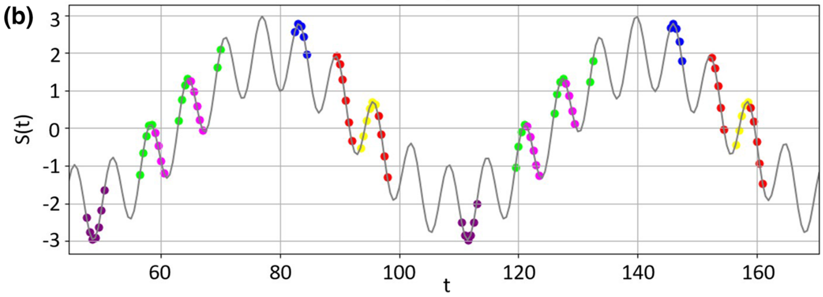


Fig. 5 Artificial dataset classified into **a** 2 different communities and **b** 19 different communities (only 6 of them are plotted). In these pictures, the communities are mapped back to the time series, verifying the temporal relation between the communities

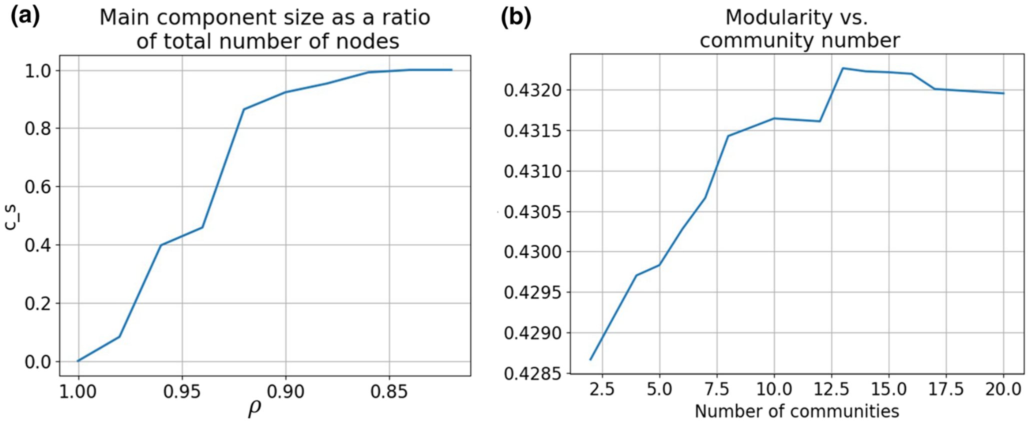


Fig. 6 Parameter used to set w and ρ for the temperature dataset. **a** Size of the main component c_s as a percentage of the total number of nodes; **b** modularity variation along the community detection process

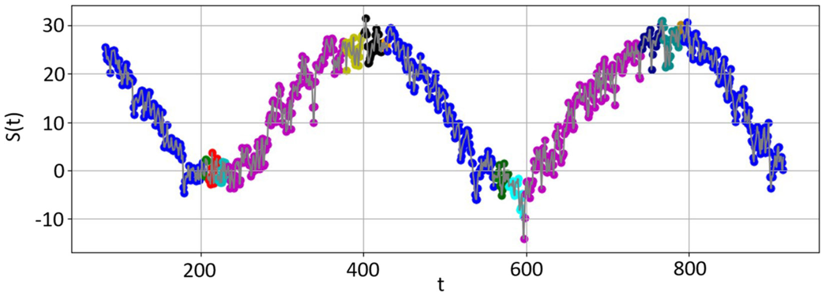


Fig. 7 Classified temperature data

The classified data can be seen in Fig. 9. Notice that other reversal patterns also appear, others than the two major trends (represented in green and purple). However, in this case, the modularity gain from 2 to 18 is even less significant than in the previous experiments, and no sub-patterns are found.

These three data-sets show how the method is able to capture the underlying patterns according to dif-

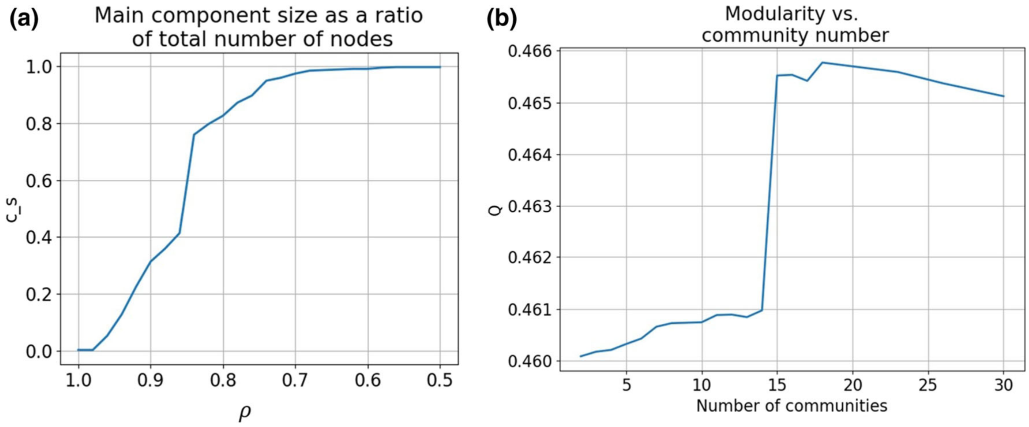
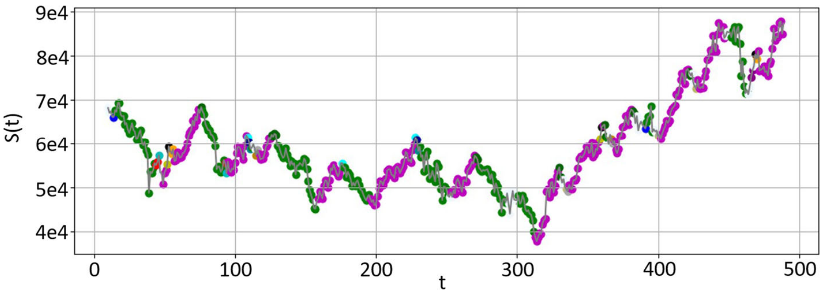


Fig. 8 Parameter used to set w and ρ for the stock price dataset. **a** Size of the main component c_s as a percentage of the total number of nodes; **b** modularity variation along the community detection process

Fig. 9 Classified stock price dataset



ferent types of data. The artificial case presents clear cycles and no noise. Therefore, the highest modularity generated meaningful patterns. The temperature data presents clear cycles but with a lot of noise. In this case, the method captured the main patterns and the reversal patterns when using the highest modularity division. Finally, the stock market data had a very low gain in modularity after the first division.

6 Conclusions

This work explored the community structure of the time series by proposing a framework to study how the patterns observed in a time series relate to the constructed network's community structure.

We showed that the resulting network's modularity can be optimized by adjusting the construction parameters and that this results in clear time series patterns. We also showed that depending on the original data, the pattern could be meaningful or not, and, therefore, the division process should be stopped in an earlier step.

One of the goals of this work was to depict how the characteristics of a time series are carried to a complex network structure and how it can be mapped back to the time series. Therefore, although we present a straightforward process for analyzing time series patterns, the idea proposed here can be extended to more complex scenarios and aimed at different objectives such as classification task and predictive models. Also, both the mapping method, which used the Pearson correlation, and the community detection method can be replaced by other methods. For example, as future work, a community detection method that captures community overlaps could be employed since many stochastic time series present overlapping patterns.

Acknowledgements This research is supported by the Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) under Grant no.: 2015/50122-0 and the German Research Council (DFG-GRTK) Grant no.: 1740/2. D.A.V.O acknowledges the computational resources from the Center for Mathematical Sciences Applied to Industry (CeMEAI) under Grant 2013/07375-0, and FAPESP Grants 2016/23698-1, 2018/24260-5, and 2019/26283-5.

Author contribution statement

L.A. and D.A.VO conceived the method and conducted the experiments. T.C.S and L.Z supervised the project. All authors discussed the ideas and the results. All authors reviewed the manuscript.

References

1. S. Rani et al. Review on time series databases and recent research trends in Time Series Mining. In: 2014 5th international conference on confluence the next generation information technology summit, pp. 109–115 (2014)
2. A. Baheti, D. Toshniwal. Trend analysis of time series data using data mining techniques. In: 2014 IEEE international congress on big data (Big Data Congress), pp. 430–437 (2014)
3. M. Small. Complex networks from time series: capturing dynamics. In :2013 IEEE international symposium on circuits and systems, pp. 2509–2512 (2013)
4. R.V. Donner et al., Recurrence-based time series analysis by means of complex network methods. Int. J. Bifurc. Chaos **21**(04), 1019–1046 (2011)
5. X. Gao et al., Temporal network pattern identification by community modelling. Sci. Rep. **10**(1), 1–12 (2020)
6. X. Gao et al., Characteristics of the transmission of autoregressive sub-patterns in financial time series. Sci. Rep. **4**, 6290 (2014)
7. R.V. Donner et al., Ambiguities in recurrence-based complex network representations of time series. Phys. Rev. E **81**(1), 015101 (2010)
8. L. Lacasa, V. Nicosia, V. Latora, Network structure of multivariate time series. Sci. Rep. **5**, 15508 (2015)
9. X. Li, X. Liu, K. Chi. Recent advances in bridging time series and complex networks. In: 2013 IEEE international symposium on circuits and systems, pp. 2505–2508 (2013)
10. M. Stephen, C. Gu, H. Yang, Visibility graph based time series analysis. PLoS One **10**(11), e0143015 (2015)
11. M. Wang et al. A new time series prediction method based on complex network theory. In: 2017 IEEE international conference on Big Data, pp. 4170–4175 (2017)
12. L. N. Ferreira, L. Zhao. Detecting time series periodicity using complex networks. In: 2014 Brazilian Conference on intelligent systems, pp. 402–407 (2014)
13. L.N. Ferreira, L. Zhao, Time series clustering via community detection in networks. Inf. Sci. **326**, 227–242 (2016)
14. L.N. Ferreira et al., Spatiotemporal data analysis with chronological networks. Nat. Commun. **11**(4036), 1–11 (2020)
15. L. Lacasa et al., From time series to complex networks: The visibility graph. Proc. Natl. Acad. Sci. **105**(13), 4972–4975 (2008)
16. R.V. Donner et al., Recurrence networks— a novel paradigm for nonlinear time series analysis. New J. Phys. **12**(3), 033025 (2010)
17. Z. Gao et al., Multivariate weighted complex network analysis for characterizing nonlinear dynamic behavior in two-phase flow. Exp. Thermal Fluid Sci. **60**, 157–164 (2015)
18. Z. Gao et al., Multiscale complex network for analyzing experimental multivariate time series. Europhys. Lett. **109**(3), 30005 (2015)
19. Z. Gao et al., Multi-frequency complex network from time series for uncovering oil-water flow structure. Sci. Rep. **5**, 8222 (2015)
20. Z. Gao, M. Small, J. Kurths, Complex network analysis of time series. Europhys. Lett. **116**(5), 50001 (2017)
21. J. Zhang, M. Small, Complex network from pseudoperiodic time series: topology versus dynamics. Phys. Rev. Lett. **96**(23), 238701 (2006)
22. S. Fortunato, Community detection in graphs. Phys. Rep. **486**(3–5), 75–174 (2010)
23. M.E. Newman, M. Girvan, Finding and evaluating community structure in networks. Phys. Rev. E **69**(2), 026113 (2004)
24. A. Clauset, M.E. Newman, C. Moore, Finding community structure in very large networks. Phys. Rev. E **70**(6), 066111 (2004)
25. G. Palla et al., Uncovering the overlapping community structure of complex networks in nature and society. Nature **435**(7043), 814 (2005)
26. A.L. Barabási et al., *Network Science* (Cambridge University Press, Cambridge, 2016)
27. P.K. Gopalan, D.M. Blei, Efficient discovery of overlapping communities in massive networks. Proc. Natl. Acad. Sci. **110**(36), 14534–14539 (2013)
28. Y. Yang, H. Yang, Complex network-based time series analysis. Phys. A **387**(5–6), 1381–1386 (2008)
29. L. Anghinoni et al., Time series trend detection and forecasting using complex network topology analysis. Neural Netw. **20**, 20 (2019)
30. S. Zhang et al., Cautionary tales on air-quality improvement in Beijing. Proc. R. Soc. A **473**(2205), 20170457 (2017)

CHAPTER

3

QUANTITATIVE ANALYSIS OF THE EFFECTIVENESS OF PUBLIC HEALTH MEASURES ON COVID-19 TRANSMISSION

Preprint, medRxiv, 2020. Silva, T. C., Anghinoni, L., & Zhao, L. (2020). Quantitative analysis of the effectiveness of public health measures on covid-19 transmission. medRxiv, 2020-05. <https://doi.org/10.1101/2020.05.15.20102988>

Author contribution statement

T.C.S., L.A. and L.Z. conceived the study idea and researched the related work. T.C.S. designed the method and conducted the experiments. T.C.S. and L.A. gathered the data-sets. L.Z. supervised the research scheme. All authors reviewed the manuscript.

Quantitative Analysis of the Effectiveness of Public Health Measures on COVID-19 Transmission

Thiago Christiano Silva*

Universidade de São Paulo, Ribeirão Preto, Brazil

Universidade Católica de Brasília, Brasília, Brazil

Leandro Anghinoni

Universidade de São Paulo, São Carlos, Brazil

Liang Zhao

Universidade de São Paulo, Ribeirão Preto, Brazil

Abstract

Although COVID-19 has spread almost all over the world, social isolation is still a controversial public health policy and governments of many countries still doubt its level of effectiveness. This situation can create deadlocks in places where there is a discrepancy among municipal, state and federal policies. The exponential increase of the number of infectious people and deaths in the last days shows that the COVID-19 epidemics is still at its early stage in Brazil and such political disarray can lead to very serious results. In this work, we study the COVID-19 epidemics in Brazilian cities using early-time approximations of the SIR model in networks. Different from other works, the underlying network is constructed by feeding real-world data on local COVID-19 cases reported by Brazilian cities to a regularized vector autoregressive model, which estimates directional COVID-19 transmission channels (links) of every pair of cities (vertices) using spectral network analysis. Our results reveal that social isolation and, especially, the use of masks can effectively reduce the transmission rate of COVID-19 in Brazil. We also build counterfactual scenarios to measure the human impact of these public health measures in terms of reducing the number of COVID-19 cases at the epidemics peak. We find that the efficiency of social isolation and of using of masks differs significantly across cities. For instance, we find that they would potentially decrease the COVID-19 epidemics peak in São Paulo (SP) and Brasília (DF) by 15% and 25%, respectively. We hope our study can support the design of further public health measures.

Keywords: COVID-19, SARS-CoV-2, health policy, network, VAR, SIR

1. Introduction

The quick spread of the COVID-19 across countries has evidenced the high degree of interconnectedness worldwide. In less than six months, the COVID-19 epicenter traveled around the globe, starting in China, then moving to Italy, and to the US. The Coronavirus Resource Center at the John Hopkins

*Corresponding author. Email: thiago.christiano.silva@usp.br. Address: Universidade Católica de Brasília, SGAN 916 Módulo B Avenida W5, CEP 70790-160, Asa Norte, Brasília, DF.

University registers more than 4 million cases of the COVID-19 spread around 187 affected countries, i.e., roughly 96% of all countries recognized by the United Nations. Factors of such a rapid spreading include large flows of international air transportation, enabling cross-country jumps of the new coronavirus. Recently, the airline industry has been experiencing large drops in revenue mainly because of international border closures implemented by governments worldwide to detain "imported transmissions" of the virus. However, COVID-19 cases still substantially grow inside borders and represent a serious health concern of several countries across the globe. In this scenario, we can say that concerns about cross-country transmission have reduced and the understanding of the COVID-19 domestic transmission has gained much relevance.

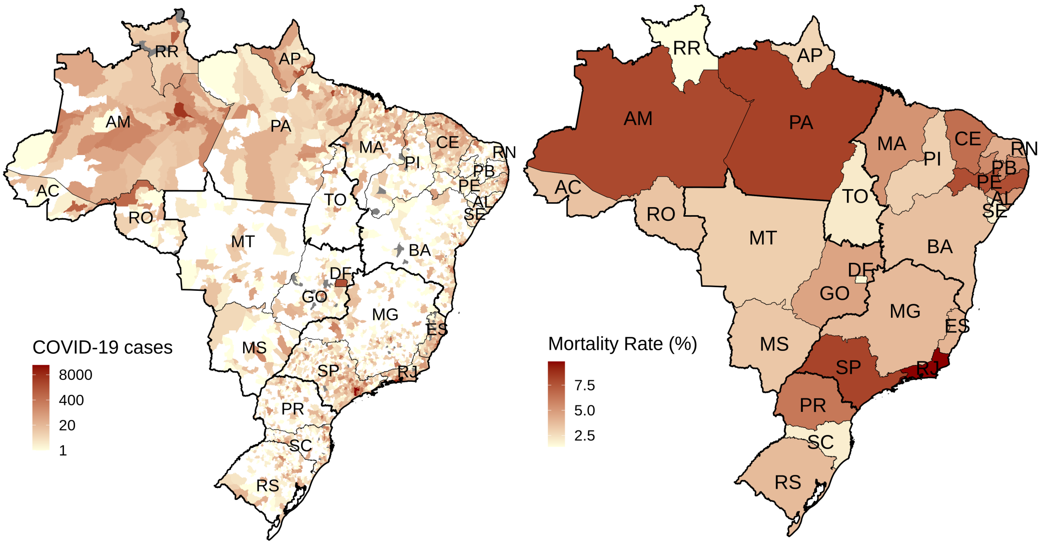
This paper focuses on the COVID-19 domestic transmission in Brazil, which already registers cases in all 27 states as depicted in Figure 1a. We analyze the efficiency of public health measures—such as social isolation/quarantine and use of masks—in mitigating the COVID-19 transmission in the country using an innovative network-based approach that accounts for intra and intercity COVID-19 transmission channels. There are several unique features that make Brazil an important case study. First, there is a political confusion about the effectiveness of social isolation by the Brazilian federal and state governments [1]. The exponential increase in the number of infectious people and deaths in the last days indicates that such political disarray can lead to very serious results. Second, Brazil contains the 6th largest population in the world. Thus, the human impact of the COVID-19 can be substantial if not properly mitigated and a second wave of cross-country spillovers could be potentially sizable in the future.[1] Third, Brazil has significant socioeconomic and cultural disparities across its 5,570 cities. Therefore, COVID-19 transmission and mortality rates may largely differ across cities, such as evidenced in Figures 1a–1b. The model proposed in this paper is able to estimate these city-specific COVID-19 transmission rates, thus accounting for their distinctive aspects. Fourth, WHO reports show that Latin America will most probably be the next epicenter of the COVID-19 outbreak. Since Brazil is the largest Latin American country and borders 83% of all South American countries, an understanding of the regional aspects of the COVID-19 transmission is crucial for designing public health measures.

Most countries in the Americas are still facing the early stages of the COVID-19 and Brazil is no different. While it is important to have a full picture of the pandemic in each country to better design government policies aimed at mitigating the COVID-19 spread considering their local particularities, the omission of the government in taking effective measures at the onset of the epidemics can have large human and economic effects in the long term. Some eastern countries, such as China, South Korea and Singapore, may be an indication that having previous organized policies and mask usage culture are key to successfully mitigate the death toll. In this work, we consider only the availability of early-time data on the COVID-19 dynamics, thus better reflecting the real-world conditions that most governments are facing.

The dynamic of the COVID-19 epidemics is not only determined by the local aspects of cities. There is a continuous flow of persons from and to different cities either through roadways, domestic airlines, or sea routes that could transport the disease. However, these intercity transmission spillovers are not limited to biological risk factors. For instance, economic activities could also be related to the propensity of acquiring the virus from other places, such as when households or firms buy supplies abroad that are conditioned on surfaces that the virus is viable for long periods without proper sanitation.[2]

[1] In [2], the authors projects recurrent wintertime outbreaks of SARS-CoV-2 will occur after the initial pandemic wave. They argue that prolonged or intermittent social distancing could be necessary up to 2022. Even with apparent elimination, the authors state that the resurgence in contagion could be possible as late as 2024.

[2] Studies have show that the virus is more stable on smooth surfaces, such as plastic and stainless steel (detectable up to 7 days), and is very sensitive to temperature (the inactivation time is reduced to 5 mins at 70 degrees Celsius)[3]. The aerosol and surface stability of SARS-CoV-2 is similar to SARS-CoV-1, with a half-life of about 1hr in the form of aerosol and up to 7hrs on plastic surfaces. Other surfaces, such as copper, cardboard and stainless steel have also long half-life values, ranging from 1 to 6 hours [4].



(a) Infectious persons (city level)

(b) Mortality rate (state level)

Figure 1: *COVID-19 geographical spreading pattern in Brazil in terms of (a) COVID-19 cases at the city level and (b) mortality rate at the state level as of May 8, 2020. Gray areas represent cities that have not reported any COVID-19 epidemiologcal bulletin. We evaluate mortality rates by taking the ratio of the number of deaths due to COVID-19 to the number of infectious persons. Mortality rates are probably upward biased because the number of observed infectious persons is likely to be underestimated, as the COVID-19 may pass unnoticed for some cases (mild or no adverse conditions at all). We report mortality rates at the state rather than city level because there are many cities with few COVID-19 cases and deaths, which would distort the estimated mortality rates.*

This transmission dynamic renders each city subject not only to its inherent "COVID-19 natural transmission rate" dictated by the local aspects of the city itself—such as demography, culture, law, and weather—but also from outside the city. Our model permits to estimate transmission rates of each city while accounting for infectious factors from the outside using the Susceptible-Infectious-Recovered (SIR) model in a special type of transmission network among cities.

We take an innovative approach to construct the underlying COVID-19 transmission network among cities. Even though we apply the model for the COVID-19 propagation inside Brazil, the model is general and could be applied for any networked environment, such as in cross-country studies or even more granular approaches than at the city level. We model such network using a weighted directed graph. Vertices are cities and links represent potential COVID-19 contagion/spillovers from one city to another. To estimate the links, we consider a panel-format data [3] composed of city-specific COVID-19 infectious counts of locals over time. We then use a vector autoregressive (VAR) model to find directional COVID-19 transmissions of every pair of cities in the network. Since the seminal paper of [5], VARs have provided key empirical input into substantive economic and financial aspects. Despite the robustness of the model, their use in epidemiology is still a new topic. Here, we design a VAR model that explicitly considers the temporal ordering of the disease spreading. We let every city-specific infectious count be dependent not only on its own past value but also from all other cities. The weights of past values of each city j that influence the current city i's infectious local count are the links in our network. Such links are estimated by fitting the entire network structure to temporal

[3] A *panel data* is composed of n multivariate time series, each representing the evolution of COVID-19 cases of a specific city. It is a mixture of *cross-sectional data*—in which we observe n cities all in a specific time point—and *time series data*—in which we observe a single individual over time.

city-specific infectious count data. We mitigate concerns with parameter overfitting by using an elastic net regularization scheme during training time[4] and one-step ahead rolling validation methodologies borrowed from the machine learning literature.[5]

An interesting property of the early-time dynamics of a SIR model is that it still enables us to estimate the transmission rate β of the model. Given the recovery rate γ of infectious persons,[6] then the model can be completely described [11], including late-time dynamics and infectious peak. It is worth mentioning that the rate γ can be divided into two parts, the time from onset to death and the time from onset to recovery. Both can vary from country to country, since they are highly correlated to demographics, health care system and the treatments available. The onset to recovery time is, however, invariant to the topological structure of the system and, therefore, we use an average value of 14 days in all scenarios of our study

In early-time dynamics, the effective transmission rate β of an *isolated SIR* and a *networked SIR* model differs by the spectrum of the estimated COVID-19 transmission network. When we do not consider the network environment, we are effectively supposing the existence of a single large city composed of all cities in the model. In this way, the susceptibility of being infected depends on the total number of infected (all cities). The introduction of multiple cities effectively reduces this propensity by imposing that the likelihood of being infected is higher inside cities rather than across cities. The network spectrum corresponds to the largest eigenvalue of the network adjacency matrix. If the isolated SIR has a transmission rate β, then the networked SIR will have an effective transmission rate of $\beta_{\text{eff}} = \lambda_{\max}\beta$, in which $\lambda_{\max}$ is the largest eigenvalue of the network. The network spectrum encodes all the graph structure in terms of its ability of spreading and amplifying intercity contagion at early time.

In this paper, we also analyze the efficiency of health policy measures implemented by the Brazilian government to mitigate the COVID-19 propagation. Social isolation and quarantine measures were adopted by several states at different time scales. Following that, the Brazilian Health Ministry recommended the use of masks at the federal level. Political disagreements on the effectiveness of quarantine measures by the federal and state governments were on display and may have lead the population into confusion, thus affecting the efficacy of such measures. Our work contributes to this discussion by estimating the joint efficacy of these measures.

We find that the quarantine and use of masks measures decreased the growth rate of the spectrum of the COVID-19 transmission network over time, suggesting that the measures were effective. To get a sense, Figure 2 portrays the average COVID-19 growth rate of cities in the state of São Paulo segregated in terms of their average social distancing index in the period.[7] First, after the use of masks recommendation, the COVID-19 growth rate, in general, decreased. However, it decreased more in cities of São Paulo with low social distancing measures. This may be due to the fact that these cities

[4] The elastic net is composed of a convex combination of the Lasso (L_1) and Ridge (L_2 norm) regularization. We refer the reader to the seminal work of [6] for further details.

[5] Parameter overfitting becomes a serious concern when we have several cities in the model. For instance, we apply our method to Brazilian data, which is a country with vast territorial dimensions and with more $5,570$ cities (end of 2019). In this case, we would have to estimate $5,570 \times 5,570 \approx 31$ million parameters with only a few time points (because we only have early-time data). Ensuring regularization is vital to have reasonable out-of-sample estimates. See [7] for more details on regularization of VAR models.

[6] The recovery rate can be estimated from the timeline between the appearance of symptoms and the case resolution. Several ongoing studies report estimates for the recovery rate. For instance, the authors in [8] assumes that the duration of the infection ranges from 15 to 20 days. Data from the outbreak in Wuhan show an onset-to-death time of 17.8 days and an onset-to-recovery time of 24.7 days [9]. This results are, however, biased to higher values due to the overwhelmed health care system in Wuhan in the early days of the outbreak and the sub-notification of the outcome of mild-cases. Reports from WHO indicate a recovery time of 14 days for mild cases and 21-42 days for severe cases. Among those who die the onset to outcome ranges from 14 to 56 days [10]. Since the mild cases account for most of the cases, we set gamma to 14 days in this study.

[7] Such index represents the extent of compliance of the population to the quarantine measures.

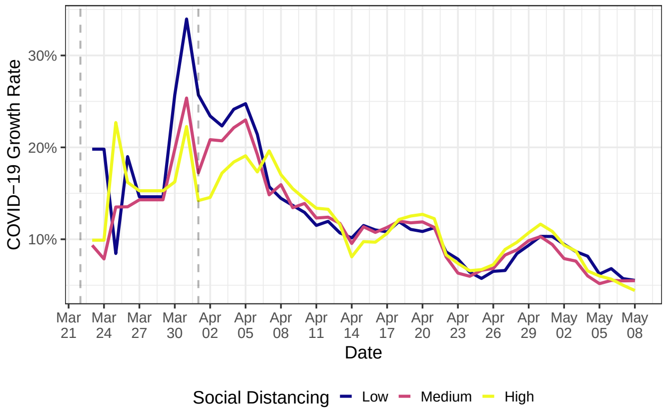


Figure 2: *Average COVID-19 growth rate in cities of the state of São Paulo, Brazil, with low, medium, and high social-distancing indices. Available data goes until May 8, 2020. The first vertical line is the beginning of SP quarantine, while the second represents the use of mask recommendation by the federal government. Data from social distancing is public and comes from the São Paulo State Government (in Portuguese). To alleviate week seasonality, we use 7-day moving averages to construct the average growth rates. The low, medium, and high social-distancing indices represent the bottom, middle, and upper terciles of the corresponding distribution. Data from the number of infectious persons per each city is discussed in Section 4.1.*

could have more potential close human-to-human contact and therefore the use of masks is crucial to detain the COVID-19 transmission. To get a sense of the human impact of such measures, we build counterfactual scenarios in which we consider that none of these measures were taken by the government. By running the SIR model in networks, we find that the quarantine and the use of masks recommendation reduced the peak of the COVID-19 epidemics, on average, in 15% in São Paulo (SP) and almost 25% in Brasília (DF), when we look at the average effect in the last week of available data (May 2 to 8, 2020). This reduction is explained by the flattening of the epidemics curve: São Paulo (SP) and Brasília (DF) have peak date shifts from July 7 to July 24 and August 29 to September 28, respectively.

Our results show the increasing trend of infectious cases in the last days, which is confirmed by the updated official data in Brazil. This situation is consistent with the decreasing social isolation rate shown by Figure 2, which, in turn, probably caused by the political discrepancy in public health measure application.

2. Related background and literature

In this section, we present relevant background on SIR models in networks and the related literature about our work.

2.1. Relevant background: early-time dynamic of SIR models in networks

In this section, we present relevant background on the Susceptible-Infectious-Recovered (SIR) model in networks. We refer the reader to [11] for a comprehensive analysis on epidemiological models and to [12] for the seminal paper on the original SIR model. Since we focus on the early-time dynamics of the SIR models, we can assume that the number of births and deaths are much smaller than the population, in a way that the closed population hypothesis holds.

Define as $\boldsymbol{s}_i(t)$, $\boldsymbol{x}_i(t)$, and $\boldsymbol{r}_i(t)$ the share of susceptible, infectious, and recovery persons of city i relative to the local population at time t. In a closed population, the SIR model in networks is government by the following differential equations:

$$
\begin{aligned}
\frac{d}{dt}\boldsymbol{s}_i(t+1) &= -\beta \cdot \boldsymbol{s}_i(t) \cdot \sum_{j \in \mathscr{V}} \boldsymbol{A}_{ij}\boldsymbol{x}_j(t) && (1)\\
\frac{d}{dt}\boldsymbol{x}_i(t+1) &= \beta \cdot \boldsymbol{s}_i(t) \cdot \sum_{j \in \mathscr{V}} \boldsymbol{A}_{ij}\boldsymbol{x}_j(t) - \gamma \cdot \boldsymbol{x}_i(t) && (2)\\
\frac{d}{dt}\boldsymbol{r}_i(t+1) &= \gamma \cdot \boldsymbol{x}_i(t) && (3)\\
1 &= \boldsymbol{s}_i(t) + \boldsymbol{x}_i(t) + \boldsymbol{r}_i(t) && (4)
\end{aligned}
$$

$\forall i \in \mathscr{V}$ and $t \geq 0$. We can substitute (4) into (2), yielding:

$$
\frac{d}{dt}\boldsymbol{x}_i(t+1) = \beta \cdot (1 - \boldsymbol{x}_i(t) - \boldsymbol{r}_i(t)) \cdot \sum_{j \in \mathscr{V}} \boldsymbol{A}_{ij}\boldsymbol{x}_j(t) - \gamma \cdot \boldsymbol{x}_i(t) \tag{5}
$$

In early time, i.e., we can assume that $\boldsymbol{x}_i(t) \ll 1$ and $\boldsymbol{r}_i(t) \approx 0$, $\forall i \in \mathscr{V}$. Therefore, we can ignore second-order $\boldsymbol{x}_i(t)$ terms and effectively set $\boldsymbol{r}_i(t)$ to 0. With these modifications, Equation (5) becomes:

$$
\begin{aligned}
\frac{d}{dt}\boldsymbol{x}_i(t+1) &= \beta \cdot \sum_{j \in \mathscr{V}} \boldsymbol{A}_{ij}\boldsymbol{x}_j(t) - \gamma \cdot \boldsymbol{x}_i(t)\\
&= \beta \cdot \sum_{j \in \mathscr{V}} \left(\boldsymbol{A}_{ij} - \frac{\gamma}{\beta}\delta_{ij}\right)\boldsymbol{x}_j(t),\\
&= \beta\left(\boldsymbol{A} - \left(\frac{\gamma}{\beta}\right)\boldsymbol{I}\right)\boldsymbol{x}(t)\\
&= \beta\boldsymbol{M}\boldsymbol{x}(t) && (6)
\end{aligned}
$$

in which $\boldsymbol{I}$ is the identity matrix, $\boldsymbol{M} = \boldsymbol{A} - \left(\frac{\gamma}{\beta}\right)\boldsymbol{I}$ is the adjacency matrix $\boldsymbol{A}$ with a homogeneous perturbation of $\frac{\gamma}{\beta}$ in the main diagonal, and $\delta_{ij} = 1$ if $i = j$, and $\delta_{ij} = 0$ otherwise. Equation (6) is a standard differential linear system whose solution can be written in terms of the eigenvector basis of the adjacency matrix $\boldsymbol{A}$:

$$
\boldsymbol{x}_i(t) = \sum_{k=1}^{V} \boldsymbol{a}_{i,k}(0)e^{(\lambda_k\beta-\gamma)t}\boldsymbol{v}_{i,k}, \tag{7}
$$

in which $\boldsymbol{A} \cdot \boldsymbol{v}_k = \lambda_k \boldsymbol{v}_k$ holds $\forall k \in \{1, \ldots, V\}$. The term λ_k is the k-th eigenvalue of $\boldsymbol{A}$, $\boldsymbol{v}_{i,k}$ is the i-th entry of the eigenvector associated with the k-th eigenvalue. The parameter $\boldsymbol{a}_{i,k}(0)$ in (7) is a scaling constant that depends on the initial condition of city i.

In early time, the growth rate of equation (7) is government by the exponent term with the largest eigenvalue $\lambda_1 = \lambda_{\max}$ of matrix $\boldsymbol{A}$, which is a well-known measure from spectral graph theory denominated graph spectrum [13]. Therefore:

$$\boldsymbol{x}_i(t) \approx \boldsymbol{v}_{i,1} e^{(\lambda_{\max}\beta - \gamma)t}, \tag{8}$$

i.e., the growth rate is $\lambda_{\max}\beta - \gamma$ and the probability of contagion is proportional to the eigenvector associate with the largest eigenvalue $\lambda_{\max}$, $\boldsymbol{v}_1$, which corresponds to the eigenvector centrality measure of the graph, according to the spectral graph theory [13].

2.2. Relative literature

Basically, there are two strategies to prevent epidemic spreading in networks [14]. One is the efficient immunization protocols and the other is to find out relevant spreaders and activation mechanisms.

Immunization strategies are methods for identification of nodes that shall be immunized, taking into account the network structure. Immunized nodes and all the incident links can be removed from the epidemic network. Immunization can not only protect immunized individuals, but can also reduce the epidemic threshold, precluding the outbreak of the disease. Among various immunization strategies, random immunization protocol is the simplest one, where a fraction of randomly selected nodes are made immune. However, in this case, the immunization threshold tends to be 1 in heterogeneous networks, indicating that almost the whole network must be immunized to suppress the disease [15]. Target immunization protocol considers special nodes to be immunized. In [16, 15], the authors show that the immunization threshold can be exponentially small over a large range of the spreading rate if considers the immunization of a fraction of nodes with the largest degree. Other approaches consider not only the critical nodes, but also the entire prevalence curve (the so-called viral conductance) [17, 18].

Although immunization is a fundamental strategy in the epidemic study, the research community pays also much attention to find out which nodes, links and local structures are most influential or effective in the spreading process [19, 20, 21, 22, 23, 24, 25, 26]. These findings aimed at understanding network measures on nodes and links, such as degree, betweenness, K-core index, closeness, link property on spreading dynamics. Besides of finding superspreaders, some researchers also worked on the identification of how topological features influence global epidemics [27, 28, 29].

However, the above mentioned strategies require the discovery of vaccine or at least partial knowledge on the epidemic network under consideration. With the mass and quick spreading of COVID-19, neither of them is a practical method to prevent the outbreak. Therefore, global intervention methods, like social isolation, even lockdown, have already been proven to be efficient. For this reason, we study the effectiveness of public intervention methods. Our results provide strong evidence on the effectiveness of public health measures, such as quarantine and use of masks, to reduce the increasing rate of infection even without detailed information of the highly dynamical population network.

3. Methodology

This section discusses the underpinnings of our methodology. Our analysis consists of the following stages:

1. *Network construction*: we construct the COVID-19 network transmission network by fitting the network links to real data.
2. *COVID-19 epidemics estimation using the SIR model*: we use the network estimated in Step 1 and simulate the COVID-19 evolution in every city of the network.
3. *Effectiveness evaluation of public health policy*: we change the network structure so as to simulate the omission of public health policies and run our epidemics model in Step 2 without the government intervention. We estimate the efficiency of the public health policies by inspecting the change in the COVID-19 epidemics peak.

3.1. Network construction using panel data

Consider the weighted directed graph $\mathscr{G} = \langle \mathscr{V}, \mathscr{E} \rangle$ in which $\mathscr{V}$ is the set of vertices and $\mathscr{E}$ is the set of links. There are $V = |\mathscr{V}|$ vertices and $E = |\mathscr{E}|$ links in the network. In our epidemiological application, vertices can represent cities, states, countries, or any well-defined entity or geographical circumscription (neighborhood, street, house etc.). For simplicity and with no loss of generality, we denominate the vertices as cities. We assume as given the set of cities/vertices $\mathscr{V}$. In contrast, links between cities i and j connote potential COVID-19 transmission from i to j and are *a priori* unknown. In the context of cities, city-to-city contagion could happen for a series of reasons, such as when infectious persons visit or migrate or even from intercity transportation of supplies covered in surfaces that the SARS-CoV-2 is viable for long periods. Therefore, the network $\mathscr{G}$ encodes all potential transmission paths between cities be through organic or non-organic media. The goal of this section is to estimate the set of links $\mathscr{E}$, i.e., the intercity COVID-19 transmission channels.

Let $\boldsymbol{x}(t) = [\boldsymbol{x}_1(t), \boldsymbol{x}_2(t), \ldots, \boldsymbol{x}_V(t)]$ denote the vector with shares of infectious persons relative to the local population of every city $i \in \mathscr{V}$ in the network at discrete time $t \geq 0$. Specifically, we denote as $\boldsymbol{x}_i(t) \in [0,1]$ the share of infectious persons within city i at time t. That is, we take the ratio between the number of infectious persons to the total local population in the city. When $\boldsymbol{x}_i(t) = 1$, then all population in the city is infectious. When $\boldsymbol{x}_i(t) = 0$, none is infectious. In-between values represent partial shares of infectious population. Define the column vector $\boldsymbol{x}_i = [\boldsymbol{x}_i(0), \boldsymbol{x}_i(1), \ldots, \boldsymbol{x}_i(T)]'$ as the COVID-19 time series evolution in city i up to time T, in which the superscript $'$ is the transpose operator. Since we perform an early-time analysis of the epidemics, T is likely to not be large. Let also the matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_V]$, $\dim(\boldsymbol{X}) = T \times V$, be all the cities' time series with the shares of infectious persons stacked in columns over all period with available data (panel data).

To construct the network, we consider the temporal ordering of the COVID-19 spread across different cities. We attempt to describe the current share of infectious persons vector $\boldsymbol{x}_t$ with the same vector immediately at the previous time step, i.e., $\boldsymbol{x}_{t-1}$ as follows:

$$\boldsymbol{x}_t = \boldsymbol{\kappa} + \boldsymbol{A} \cdot \boldsymbol{x}_{t-1} + \boldsymbol{\epsilon}_t, \tag{9}$$

$\forall t \in \{0, 1, \ldots, T\}$. The term $\boldsymbol{\kappa}$, $\dim(\boldsymbol{\kappa}) = V \times 1$, is an intercept column vector; $\boldsymbol{A}$, $\dim(\boldsymbol{A}) = V \times V$, is the adjacency matrix encoding the set of links $\mathscr{E}$ of the graph; and $\boldsymbol{\epsilon}_t \sim (\boldsymbol{0}, \boldsymbol{\Sigma}_{\boldsymbol{\epsilon}})$ is the unobservable zero mean white noise vector process (serially uncorrelated or independent) with time-invariant covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{\epsilon}}$. Let $\boldsymbol{A}_{ij}$ be the (i,j)-entry of $\boldsymbol{A}$, $i, j \in \mathscr{V}$. When $\boldsymbol{A}_{ij} > 0$, then city i can spillover COVID-19 to city j. The larger $\boldsymbol{A}_{ij}$ is, the stronger is such contagion. Then, the set of links is given by $\mathscr{E} = \{i, j \in \mathscr{V} : \boldsymbol{A}_{ij} > 0\}$.

The terms $\boldsymbol{\kappa}$, $\boldsymbol{A}$ in Equation (9) are unknown and are estimated using a fitting process to the observed data $\boldsymbol{X}$.Equation (9) describes a VAR(1) model. To ensure that the system is stable, the companion matrix must have roots inside the complex unit circle. To guarantee such property, our variables $\boldsymbol{x}_i, i \in \mathscr{V}$, must be stationary. Since they are lower- and upper-bounded—i.e., $\boldsymbol{x}_i \in [0,1]$—then they are stationary by construction. Specifically, we minimize the following regularized loss function L [7] using the coordinate descent algorithm [30]:

$$\begin{aligned} L &= \min_{\boldsymbol{\kappa}, \boldsymbol{A}} \sum_{t=0}^{T} \|\boldsymbol{\epsilon}_t\|_F^2 + \text{Regularization}\,(\boldsymbol{A}) \\ &= \min_{\boldsymbol{\kappa}, \boldsymbol{A}} \sum_{t=0}^{T} \|\boldsymbol{y}_t - (\boldsymbol{\kappa} + \boldsymbol{A}\boldsymbol{x}_{t-1})\|_F^2 + \lambda\,(\alpha\,\|\boldsymbol{A}\|_1 + (1-\alpha)\,\|\boldsymbol{A}\|_2)\,, \end{aligned} \tag{10}$$

in which $\lambda \geq 0$ is the *elastic net* regularization term and $\alpha \in [0,1]$ is the tradeoff parameter between Lasso (L_1 norm) and Ridge (L_2 norm) regularizations. We notate $\|.\|_F$, $\|.\|_1$, $\|.\|_2$ as the Frobenius, L_1, and L_2 norms, respectively. Larger values of λ encourage sparser networks. The first term represents minimization of the error term $\boldsymbol{\epsilon}_t, \forall t \in 0,1,\ldots,T$, and ensures that the estimated adjacency matrix $\boldsymbol{A}$ better reflects the COVID-19 transmission dynamics over time. The second term is a regularization term over the adjacency matrix $\boldsymbol{A}$ introduced to prevent overfitting and ensure that the estimation is numerically tractable. We do not regularize the intercept vector $\boldsymbol{\kappa}$ because it conceptually adapts to the city-specific average values of our data.

There is an empirical challenge in fitting the adjacency matrix $\boldsymbol{A}$ to the panel data $\boldsymbol{X}$ when we are dealing with large-scale networks in which the number of cities V largely surpasses the number of available time points T, i.e., when $V \gg T$. Such problem is aggravated when we only have early-time information about the disease, i.e., T is small. In this case, we would incur in overparametrization and overfitting is a concern. The regularization term in (10) mitigates such concern. We opt for an *elastic net* regularization scheme because it is a robust regularizator that combines positive features of Lasso and Ridge regularizations [30].

Due to the temporal dependency of the panel data, the usual k-fold cross-validation is not well-suited for our model selection procedure. Following [7], we optimize the penalty parameters λ and α in (10) using a h-step ahead mean-square forecast error (MSFE). Due to data availability, we keep $h = 1$ so as to minimize further data losses. We divide the data into three equally-spaced and contiguous periods: (i) initialization ($t \in \{0,\ldots,T_1\}$), (ii) training ($t \in \{T_1+1,\ldots,T_2\}$), and (iii) forecast evaluation ($t \in \{T_2+1,\ldots,T\}$), in which $T_1 = \lfloor \frac{T}{3} \rfloor$ and $T_2 = \lfloor \frac{2T}{3} \rfloor$. We also use a rolling validation process as follows. We first fit the model using all data up to time T_1 and forecast $\hat{\boldsymbol{x}}_{T_1+1}^{(\lambda_c,\alpha_c)}$, in which λ_c and α_c are fixed candidate penalty terms. We then sequentially add one observation at a time and repeat this process until $T_2 - 1$. Then, we choose the penalty terms λ and α that minimize the one-step ahead MSFE given by:

$$MSFE(\lambda,\alpha) = \frac{1}{T_2 - T_1} \sum_{t=T_1}^{T_2-1} \left\| \hat{\boldsymbol{x}}_{t+1}^{(\lambda,\alpha)} - \hat{\boldsymbol{x}}_{t+1} \right\|_F^2. \tag{11}$$

Finally, we estimate the one-step ahead forecast accuracy using data points in $t \in \{T_2,\ldots,T\}$, which have not been used in the model selection procedure. To better assess the potentiality of the network in amplifying contagion across different municipalities, we remove the self-loops in the estimated network, which correspond to the influence of the local infectious population on its own future value.

3.2. Estimating transmission rate in early-time epidemics networks

In this section, we assume the network structure $\mathscr{G} = \langle \mathscr{V}, \mathscr{E} \rangle$ as given, i.e., the set of vertices and links are already established in accordance with the network construction described in Section 3.1. We start from the results of the early-time dynamic of SIR models in networks described in Section 2.1. Therein, we show that the growth rate at early time is determined by $\lambda_{\max}\beta - \gamma$ (see Equation (8)). Therefore, the graph spectrum $\lambda_{\max}$ modulates the transmission rate parameter by either amplifying or dampening the contagion speed.

If $\lambda_{\max}\beta > \gamma$, then Equation (8) grows exponentially, while it decays when $\lambda_{\max}\beta < \gamma$. Therefore, the reproduction number (critical point) is $R_0 = \frac{\lambda_{\max}\beta}{\gamma}$. Recall that the reproduction number in the SIR model without network is $R_0 = \frac{\beta}{\gamma}$ [12]. Therefore, the reproduction numbers of both models differ by the graph spectrum of $\boldsymbol{A}$, $\lambda_{\max}$.

Equation (8) assumes that every city in the model has a single growth rate dynamics dictated by the term

$\lambda_{\max}\beta - \gamma$. Changes in the epidemics spreading for each city would then be fully determined by their eigenvector centralities, because growth rates are identical across cities (see Equation (8)). However, studies show that the transmission rate parameter β is dependent on local aspects of cities [11]. In contrast, the recovery rate parameter is much less variable across different places. As mentioned earlier in this study, WHO indicates an average recovery time of 14 days for mild cases. Therefore, we consider a different transmission rate for each city in the network β_i while letting fixed the recovery rate γ for all cities. We can still apply the classical framework of SIR in networks because, even though transmission rates are city specific, they tend to be normally distributed around some mean natural value. That is, large deviations are unusual. We empirically find this fact using our application to the Brazilian case. Mathematically, we rewrite (8) as follows:

$$\boldsymbol{x}_i(t) \approx \boldsymbol{v}_{i,1} e^{(\lambda_1 \beta_i - \gamma)t}. \tag{12}$$

We can linearize (12) by simply taking the $\log(.)$ at both sides of the equation for each city i in the network:

$$\log(\boldsymbol{x}_i(t)) = \log(\boldsymbol{v}_{1,i}) + (\lambda_1 \beta_i - \gamma)t, \tag{13}$$

$\forall i \in \mathscr{V}$. The LHS and RHS are always non-negative, because $\boldsymbol{x}(0) \geq 0$ and is non-decreasing (early-time assumption), $e^{(\lambda_1 \beta_i - \gamma)t} \geq 0$ (asymptotically speaking), and $\boldsymbol{v}_{1,i} \geq 0$ [13]. We can then apply the $\log(.)$ without any restrictions. We can estimate (13) for all cities i at once by adding dummies for the constant and time-dependent term for each city in the model (2 dummies per each city). We end up with a set of $2n-1$ dummy variables, because the last one is the reference dummy. Since we have a panel data with temporal dependencies (the same city appears multiple times), we use a linear panel-data estimation model [31] as follows:

$$\boldsymbol{x}_i(t) = \sum_{j \in \mathscr{V}} \delta_{ij} \left[\alpha_j + \rho_j \cdot t\right] + \epsilon_i(t), \tag{14}$$

$\forall i \in \mathscr{V}$, in which α_i and ρ_i are the constant and time-variant dummy terms for city i, and $\varepsilon_i(t)$ is the residual from the least square estimation with dummies. We cluster the errors at the city level, such as to mitigate concerns with heteroskedasticity and serial correlation, which could bias our coefficient estimates. Equations (13) and (14) are linked by the following identities:

$$\alpha_i = \log(\boldsymbol{v}_{1,i}) \Rightarrow \boldsymbol{v}_{1,i} = e^{\alpha_i}, \tag{15}$$

$$\rho_i = \lambda_1 \beta_i - \gamma \Rightarrow \beta_i = \frac{\rho_i + \gamma}{\lambda_{\max}}.. \tag{16}$$

Given the recovery rate γ—which is assumed to not change over time nor across cities—we can fully identify the eigenvector centrality and the local transmission rate of every city i using (15) and (16), respectively. We only take city-specific estimations of $\boldsymbol{v}_{1,i}$ and β_i that are statistically significant at the 10% level. Otherwise, we set the estimated coefficients to zero.

3.3. Assessing efficiency of health policy measures in epidemics spreading

With our framework, we can analyze the speed of the epidemics spreading through the network at early time by simply inspecting the graph spectrum $\lambda_{\max} = \lambda_1$ for different time horizons using the methodology described in Section 3.1. Since the reproduction number of the epidemics is proportional to the graph spectrum, then large graph spectra indicate a higher speed of contagion. Any changes of the graph spectrum can be attributed to a "net effect" of public policies of the government in the entire network. Since we use the share of infectious persons of each Brazilian city, then this "net effect" comprises not only federal policies, but also state- and even city-level policies.

Moreover, we can estimate the human impact of these policies in terms of changes in the number of infectious persons at the peak by running the SIR model described in Section 3.2 for each estimated city-specific transmission rate parameter β_i defined in (16) and for different values of the graph spectrum. We use a conservative approach and compare the largest observed graph spectrum with the most recent graph spectrum in our dataset. We assume that the largest graph spectrum occurs when public policies were still latent and were not having effects in the epidemics spreading. Most recent values of the graph spectrum are assumed to represent transmission dynamic after public policies were in, as was the case in Brazil who adopted quarantine and recommended the use of masks in the period that we have available data.

4. Application

In this section, we apply our model to Brazilian data at the city level.

4.1. Data

We use daily data on the number of infectious persons per each city in Brazil using COVID-19 epidemiological bulletins of 27 State Health Departments from February 25, 2020, to May 8, 2020.[8] Each Brazilian state compiles local reports from cities inside their geographical circumscription. We end up with 60,021 city-time epidemiological bulletins comprising 2,754 (out of 5,570) cities affected by COVID-19 in Brazil.

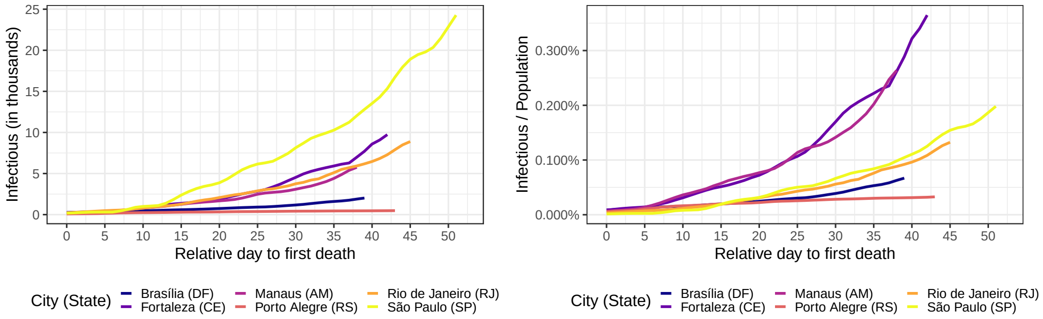
Our data is representative because local hospitals are required by law to register any COVID-19 events to the local government while cities and states must notify the federal government. However, there may be substantial sub-notifications due to persons that acquire the COVID-19 and recover unnoticed or without hospitalization.

We also collect city-level population estimates in the Brazilian Institute of Geography and Statistics (IBGE), which is the agency responsible for official collection of statistical, geographic, cartographic, geodetic and environmental information in Brazil. We evaluate the share of infectious persons by taking the ratio of COVID-19 cases reported in the local health bulletin and the local population size. The use of shares in our estimation models is important because it is a stationary variable.

We apply a three-day smoothing filter on the number of infectious persons in each municipality to alleviate concerns with late contamination reports or short-term rectifications by the local health government that could compromise our estimations. In our network construction procedure (see Section 3.1), we keep only cities that reported COVID-19 cases in at least 20% of the available time frame. Our results remain qualitatively the same if we do not apply this filtering criterion. In our estimation of the SIR parameters (see Section 3.2), we center all time points in relation to the occurrence of the first death in the city.

[8] This data is scattered around a large quantity of state government sites. In general, the bulletins are not standardized across different states and not even cities. We use the compiled dataset from Brasil.io for this task.

[8] Asymptomatic and mild-cases can represent up to 80% of the cases according to China reported numbers. This cases tend not to be tested in Brazil.



(a) Infectious persons

(b) Infectious persons / local population

Figure 3: *COVID-19 evolution in six of the most affected cities in Brazil (a) in absolute terms (number of infectious persons) and (a) as a share of the local population size. Horizontal axis represent the relative day in terms of the first observed death due to the COVID-19.*

Figures 3a–3b portray the COVID-19 evolution in six of the most affected cities in Brazil relatively to the first reported death in terms of the number of COVID-19 cases and as a share of the local population size, respectively. São Paulo (SP) has the highest number of infectious persons. However, there is strong size effect: São Paulo (SP) has almost 12.2 million residents while the second largest city, Rio de Janeiro (RJ), has almost half of that (6.8 million). To get a sense of the local COVID-19 criticality, we can look at its evolution as a share of the local population. In this case, we note that COVID-19 transmission speed is much larger in Manaus (AM) and Fortaleza (CE). Brasília (DF) and Porto Alegre (RS) have smaller transmission rates and local COVID-19 criticality. However, mortality rates may not follow such incidence criticality, because they correlate with local health quality and demography characteristics.

4.2. Results

This section presents the main empirical results of the paper. We first build the COVID-19 intercity transmission network and analyze its propensity of amplifying the COVID-19 in different cities. Then, we analyze the net effectiveness of public health measures adopted by the Brazilian government.

4.2.1. Intercity COVID-19 transmission network in Brazil

Figure 4 shows the graph spectrum of the COVID-19 intercity transmission network of Brazil over time. For each time point (horizontal axis), we run the network construction through the fitting process in Section 3.1 with data from the beginning of the sample up to that specific time point. Even though our sample starts in February 25, 2020, we start the fitting process from March 13, 2020, such as to have enough data for the fitting process. That is, we start with 18 time points for each Brazilian city. Therefore, we initially divide the panel data in three equally-sized groups with 6 time points for model training, model selection (parameters and penalty terms), and model evaluation. These group sizes increase as we add more time points. We perform the network construction estimation daily from March 13 to May 8, 2020, in an independent manner.

In Figure 4, we add a shaded area indicating the timing window in which quarantine measures were adopted by the most affected Brazilian states. Since São Paulo is the COVID-19 epicenter in Brazil as it encloses 57.4% of all the COVID-19 infections in Brazil, we also add a vertical dashed red line indicating the beginning of the quarantine adopted by the São Paulo State Government. We also draw the use of masks recommendation beginning date by the Federal Health Ministry in Brazil as a dashed blue line enacted. While quarantine measures are at the state level, the use of masks recommendations goes at the federal level and encompasses all the 5,570 cities and 27 states in Brazil. São Paulo is the most central city in the transmission network. Therefore, it practically shapes the graph spectrum of the intercity transmission network.

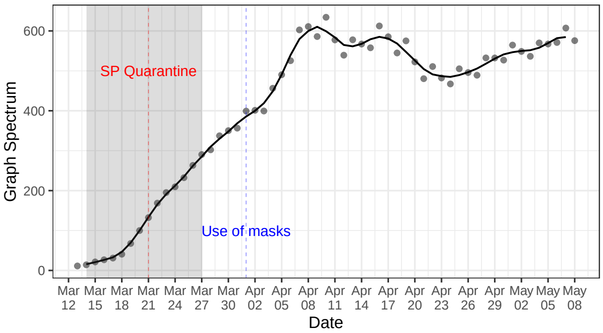


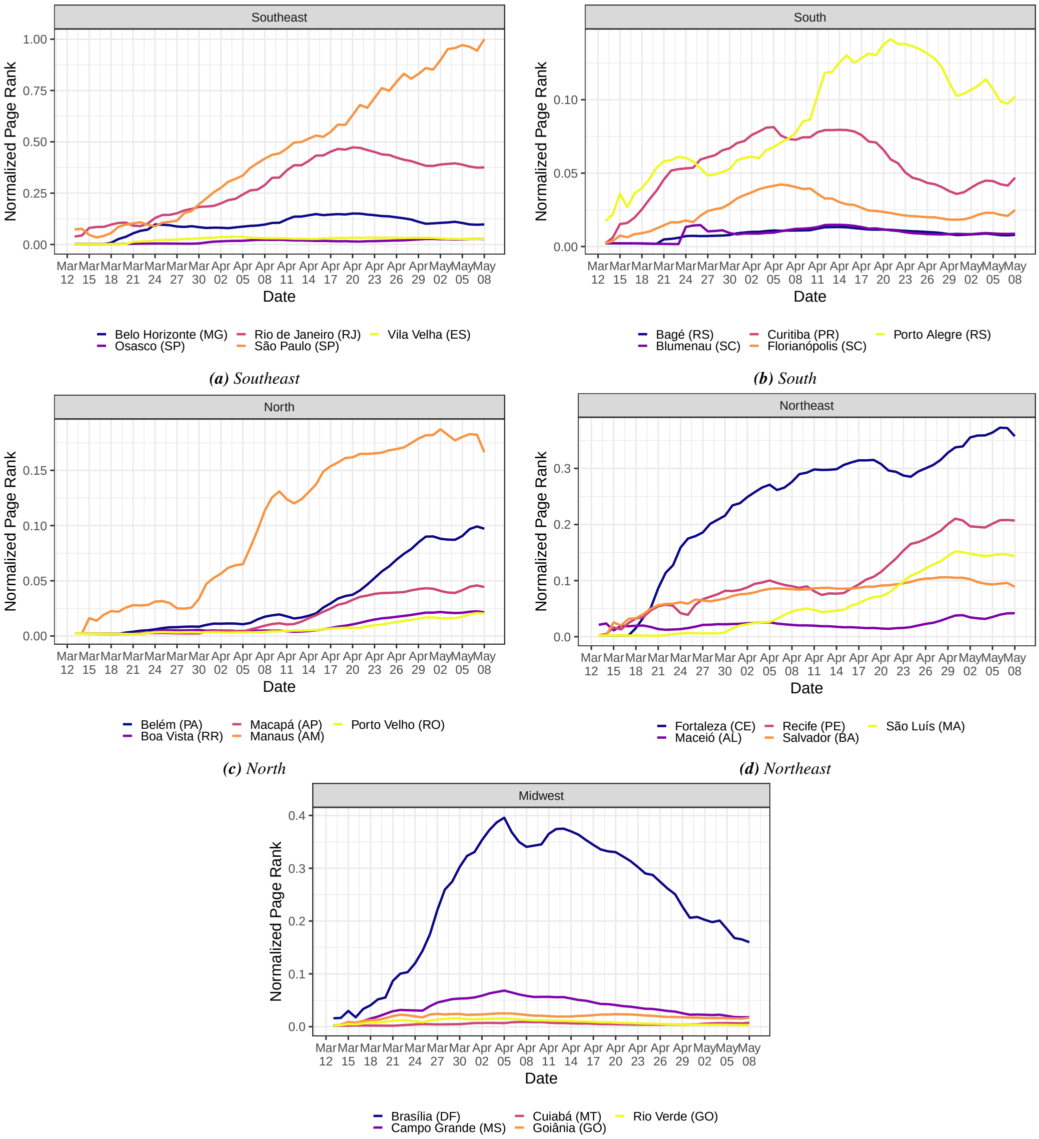
Figure 4: *Graph spectrum of the COVID-19 intercity transmission network of Brazil (see Section 3.1). The shaded area indicating the timing window in which quarantine measures were adopted by the most affected Brazilian states. The red dashed line indices the beginning of the quarantine in São Paulo, the COVID-19 epicenter in Brazil. The blue dashed line indices the beginning of the use of masks recommendation by the Federal Health Ministry. For each time point (horizontal axis), we build the network with city-specific shares of infectious persons with data up to that point.*

We observe a reduction in the growth rate of the graph spectrum after the quarantine measures precisely two days after the measure. However, the growth rate still persisted at positive rates, indicating that the COVID-19 transmission speed kept increasing after such measure, but with a slower pace. After the incubation period following the use of masks recommendation, we observe a drastic change in the graph spectrum. The growth rate changed sign and started to reduce, showing that the set of health policy measures taken by the government was efficient. However, after April 23, 2020, the graph spectrum again started to increase. This can be due to several factors, such as social confusion in following health guidelines in view of the political disarray that Brazil is facing, or even non-compliance with quarantine and use of masks measures. Our model does not permit to have an isolated causal impact of the use of masks recommendation nor of the quarantine measures. However, it enables us to understand how the set of all policy measures affected the COVID-19 transmission rate across cities over time. Combining Figures 2 and 4, it seems that the reduction in the COVID-19 growth rates after the use of masks recommendation was more apparent in cities with relative low social distancing indices. This may be due to the fact that these cities have more potential close human-to-human contact and therefore the use of masks is crucial to detain the COVID-19 transmission.

To understand the topological aspects of the COVID-19 intercity transmission network, Figure 5 plots the PageRank centrality for the top 5 most central cities in each of the five regions in Brazil. We normalize the PageRank with respect to the most central city: São Paulo (SP) on May 8, 2020. As the city centrality becomes higher, the more it contributes to spreading the COVID-19 throughout the network. The top 5 most central cities in the country are the following state capitals (in decreasing order): (i) São Paulo (SP), (ii) Rio de Janeiro (RJ), (iii) Fortaleza (CE), (iv) Recife (PE), and (v) Manaus (AM). These cities all have airports and are strongly interconnected to the remainder of cities in Brazil through roadways and are likely to be the hubs for the COVID-19 spread to other nearby cities in Brazil, especially countryside municipalities. The centrality of São Paulo (SP) in the Southeast monotonically increases over the entire sample. The same roughly occurs with Manaus (AM) in the North, Fortaleza (CE) in the Northeast. Porto Alegre (RS) in the South and Brasília (DF) in the Midwest have the highest centralities in their region but with a negative growth rate in the last days of the sample. Overall, there is a very heterogeneous profile of the city centralities over time, showing the underlying non-trivial patterns in the COVID-19 transmission network.

4.2.2. Measuring the human impact of health policy measures to mitigate the COVID-19 propagation

In this section, we run the SIR in networks (see Equations (1)–(4)) with different transmission rate parameters for each city in Brazil, in accordance with (16). We first estimate the city-specific ρ_i using the panel-data information on counts of the share of infectious persons in each city in Brazil via (14).



***(a)** Southeast*

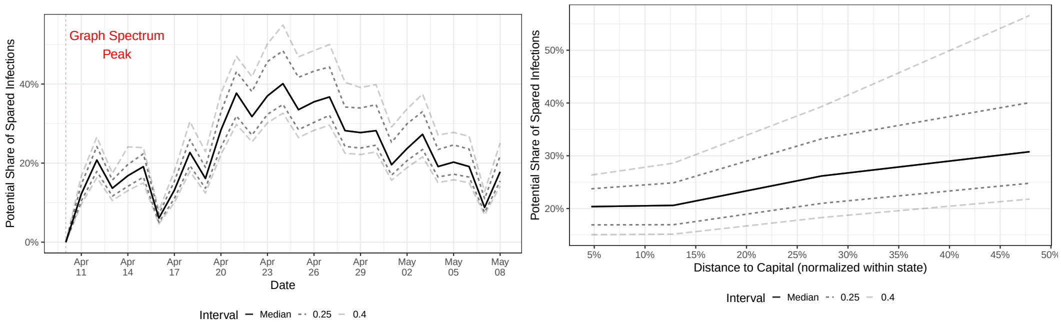
***(b)** South*

***(c)** North*

***(d)** Northeast*

***(e)** Midwest*

Figure 5: *Evolution of the normalized PageRank centrality measure in the COVID-19 transmission network (see Section 3.1 for the network construction details). We only report the top 5 cities with highest PageRank at each Brazilian region. For each time point (horizontal axis), we build the network with city-specific shares of infectious persons with temporal data up to that point. Each label is composed of the city name followed by its state inside parentheses.*



(a) *Potential share of spared infections at the peak over time*

(b) *Potential share of spared infections at the peak as a function of the city's distance to the capital (normalized within state)*

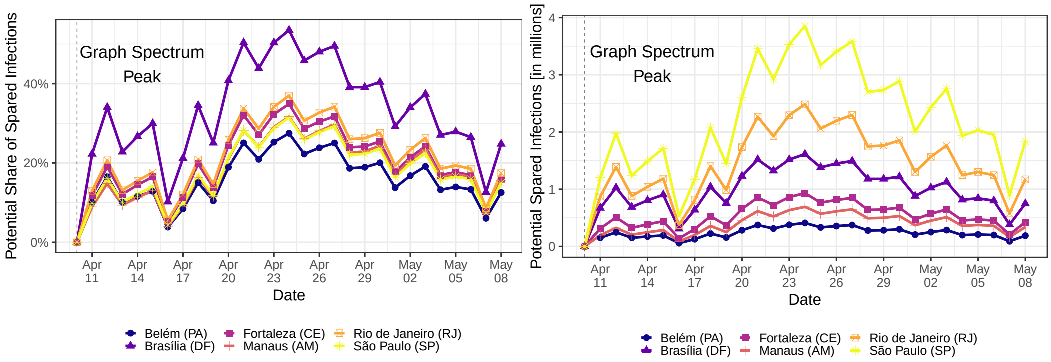
Figure 6: *Distribution of the efficiency of health policy measures along all affected cities in Brazil over time. We plot the efficiency distribution as a function of (a) time and (b) the city's distance to the capital within the same state it resides. Since states in Brazil have substantial differences in their sizes, we normalize the city's distance to the capital to the most distant city within the state.*

Then, we estimate the transmission rate parameter β_i of each city $i \in \mathscr{V}$ in Brazil by fixing the recovery rate parameter as $\gamma = 1/14$. We use the remaining parameter $\lambda_{\max}$—the graph spectrum—to evaluate the effectiveness of the set of health policy measures in detaining the COVID-19 in Brazil. We take as baseline model the graph spectrum reached in April 10, 2020, which is the maximum observed value. We assume that this graph spectrum would have not changed afterwards in case the set of health policy measures were not taken.[9] We then run several SIR models with the observed graph spectrum values in Figure 4 after the graph spectrum maximum in April, 10, 2020.

Figure 6a shows a comparison of the infectious peaks of the baseline SIR model—i.e., the hypothetical scenario in which health policy measures were not introduced—and the ones with graph spectrum values observed daily after that maximum. The vertical axis shows the relative change in these infectious peaks of the baseline and the observed model day by day, which can be interpreted in terms of the potential share of spared infections at the infectious peak due to the introduction of the set of health policy measures. Since we have data from each city affected by the COVID-19, we plot the median, percentiles 75% (0.25 distant from the median) and 90% (0.40) of this distribution. In the Supplementary Material, we provide the effectiveness of public health policies for each affected municipality in Brazil. In April 10, 2020, the share of spared infections in the epidemics peak is zero, because the baseline model is compared with itself. Then, as we move forward in time and use smaller graph spectrum values, as shown in Figure 4, the potential share of spared infectious increases. The share of spared infectious persons in the epidemics peak reaches a median value 40% lower than that of the baseline model when we use the graph spectrum in April, 24, 2020, suggesting high effectiveness of the quarantine and use of masks health policies. After this point, the share of spared persons decreases—reflecting the increase in the graph spectrum in Figure 4—giving more room for the spread of the COVID-19. The effectiveness of the health policy measures, however, remains positive throughout the entire sample.

The first case of the COVID-19 in Brazil was reported in São Paulo (SP) on February 25, 2020. After that, it spread to several Brazilian state capitals probably through air transportation (most of the airports in Brazil are in the state capitals and capitals are far from each other). The epidemics took some time before reaching the first case in countryside cities. Figure 6b displays the distribution of the potential

[9] This is a conservative approach, because we can observe a positive momentum of the graph spectrum growth rate prior to reaching April 10, 2020. However, we cannot be sure whether such graph spectrum would still increase if these policies were not in place. Therefore, we keep the conservative approach and consider such point as the maximum.



(a) Potential reduction of the share of infectious persons at the peak

(b) Potential spared persons from acquiring the SARS-CoV-2 at the peak

Figure 7: *Efficiency of public health measures over time as a function of (a) the share of the spared local population and (b) the spared number of persons (in millions). We depict curves only for six capital cities that are being substantially affected by the COVID-19: Belém (PA), Fortaleza (CE), Rio de Janeiro (RJ), Brasília (DF), Manaus (AM), and São Paulo (SP).*

share of spared infections in terms of the city distance to the state capital. Since Brazilian states are very different in size, we normalize the distance to the most distant city within the state. We observe a positive relationship between potential share of spared infectious and distance to the capital, suggesting that health public policies are most effective in cities that are distant from the capital. This may reflect not only the temporal delay of the COVID-19 in reaching the countryside, which puts the local COVID-19 at very early time in these regions, but also demography aspects, such as lower population density, and agricultural economic activities that do not require large conglomerates of persons.

Figure 7a shows the effectiveness of the set of public health measures for six of the most affected Brazilian capitals. In particular, Brasília (DF) reaches a 50% lower share of infectious persons at the peak when we compare peaks reached with the graph spectrum value on April 24, 2020 (against the baseline in April 10, 2020). Figure 7b shows the number of potential spared infectious persons due to the set of health policy measures. This figure is constructed by simply multiplying the share of spared infectious with the local population size of each of the six cities. Since São Paulo (SP) is the largest city, it would potentially spare more persons when the COVID-19 epidemics reach its peak.

5. Conclusions

At the current stage of the COVID-19 infection, many countries have stopped the entrance of foreigners. Therefore, the study of virus transmission dynamics inside each country gains relevance. In the last few days, Brazil has become one of the most infectious countries in the world. In this work, we present a general epidemics transmission model and apply it to the Brazilian case. Our method has three steps. First, we construct the COVID-19 transmission network by fitting city-specific COVID-19 cases over time to calibrate the network links, which represent intercity COVID-19 transmission. Second, we gauge the network propensity of spreading COVID-19 throughout the cities using a spectral graph analysis. Third, we propose a methodology to quantify the effectiveness of public health policies using the dynamics of early-time SIR model and spectral network theory.

Our spectral network analysis indicates that social isolation and the use of masks can effectively reduce the transmission rate of the COVID-19 in Brazil. The COVID-19 propagation dynamics seems to decrease following these public health policies when we also consider an incubation period, which lags the effect of any COVID-19 mitigation measure. Moreover, our empirical analysis supports the view that use of masks seems to be more effective than social isolation, which is further corroborated by what is being occurring in Austria [32]. With no vaccine up to date, public health intervention is still the main method of epidemic control. We hope our study can help the government make correct decisions.

Acknowledgements

This work is supported in part by the São Paulo State Research Foundation (FAPESP) under grant numbers 2015/50122-0 and 2013/07375-0, the Brazilian Coordination of Superior Level Staff Improvement (CAPES), the Pro-Rectory of Research (PRP) of University of São Paulo under grant number 2018.1.1702.59.8, and the Brazilian National Council for Scientific and Technological Development (CNPq) under grant numbers 303199/2019-9, 308171/2019-5, and 408546/2018-2.

Authors contributions statement

T.C.S., L.A. and L.Z. conceived the study idea and researched the related work. T.C.S. designed the method and conducted the experiments. T.C.S. and L.A. gathered the datasets. L.Z. supervised the research scheme. All authors reviewed the manuscript.

Declaration of interests

The authors declare no competing interests.

References

[1] The Lancet, COVID-19 in Brazil: “so what?”, 2020. The Lancet Editorial.

[2] S. M. Kissler, C. Tedijanto, E. Goldstein, Y. H. Grad, M. Lipsitch, Projecting the transmission dynamics of SARS-CoV-2 through the postpandemic period, Science (2020) eabb5793.

[3] A. Chin, J. Chu, M. Perera, K. Hui, H.-L. Yen, M. Chan, M. Peiris, L. Poon, Stability of SARS-CoV-2 in different environmental conditions, The Lancet (2020). DOI: 10.1016/S2666-5247(20)30003-3.

[4] N. van Doremalen, T. Bushmaker, D. H. Morris, M. G. Holbrook, A. Gamble, B. N. Williamson, A. Tamin, J. L. Harcourt, N. J. Thornburg, S. I. Gerber, J. O. Lloyd-Smith, E. de Wit, V. J. Munster, Aerosol and surface stability of SARS-CoV-2 as compared with SARS-CoV-1, New England Journal of Medicine 382 (2020) 1564–1567.

[5] C. Sims, Macroeconomics and reality, Econometrica 48 (1980) 1–48.

[6] H. Zou, T. Hastie, Regularization and variable selection via the elastic net, Journal of the Royal Statistical Society: Series B 67 (2005) 301–320.

[7] W. B. Nicholson, D. S. Matteson, J. Bien, VARX-L: Structured regularization for large vector autoregressions with exogenous variables, International Journal of Forecasting 33 (2017) 627–651.

[8] A. Remuzzi, G. Remuzzi, COVID-19 and Italy: what next?, The Lancet (2020).

[9] R. Verity, L. C. Okell, I. Dorigatti, P. Winskill, C. Whittaker, N. Imai, G. Cuomo-Dannenburg, H. Thompson, P. G. Walker, H. Fu, et al., Estimates of the severity of coronavirus disease 2019: a model-based analysis, The Lancet Infectious Diseases (2020). DOI: 10.1016/S1473-3099(20)30243-7.

[10] World Health Organization, China joint mission on coronavirus disease 2019 (COVID-19), https://www.who.int/docs/default-source/coronaviruse/who-china-joint-mission-on-covid-19-final-report.pdf, 2020. Online, Accessed: 2020-05-05.

[11] M. J. Keeling, K. T. Eames, Networks and epidemic models, Journal of the Royal Society Interface 2 (2005) 295–307.

[12] W. O. Kermack, A. G. McKendrick, A contribution to the mathematical theory of epidemics, Proceedings of the Royal Society of London. Series A 115 (1927) 700–721.

[13] F. R. Chung, F. C. Graham, Spectral graph theory, American Mathematical Society, 1997.

[14] R. Pastor-Satorras, C. Castellano, P. Van Mieghem, A. Vespignani, Epidemic processes in complex networks, Reviews of Modern Physics 87 (2015) 925–979.

[15] R. Pastor-Satorras, A. Vespignani, Immunization of complex networks, Physical Review E 65 (2002) 036104.

[16] R. Pastor-Satorras, A. Vespignani, Epidemic spreading in scale-free networks, Physical Review Letters 86 (2001) 3200–3203.

[17] M. Youssef, R. Kooij, C. Scoglio, Viral conductance: Quantifying the robustness of networks with respect to spread of epidemics, Journal of Computational Science 2 (2011) 286–298.

[18] P. V. Mieghem, The viral conductance of a network, Computer Communications 35 (2012) 1494–1506.

[19] D. Chen, L. Lü, M.-S. Shang, Y.-C. Zhang, T. Zhou, Identifying influential nodes in complex networks, Physica A 391 (2012) 1777–1787.

[20] D.-B. Chen, R. Xiao, A. Zeng, Y.-C. Zhang, Path diversity improves the identification of influential spreaders, Europhysics Letters 104 (2013) 68006.

[21] F. Bauer, J. T. Lizier, Identifying influential spreaders and efficiently estimating infection numbers in epidemic models: A walk counting approach, Europhysics Letters 99 (2012) 68007.

[22] L. Hébert-Dufresne, A. Allard, J.-G. Young, L. J. Dubé, Global efficiency of local immunization on complex networks, Scientific Reports 3 (2013) 2171.

[23] A. Zeng, C.-J. Zhang, Global efficiency of local immunization on complex networks, Physics Letters A 377 (2013) 1031–1035.

[24] P. Holme, T. Takaguchi, Time evolution of predictability of epidemics on networks, Physical Review E 91 (2015) 042811.

[25] R. Yi-Run, L. Song-Yang, X. Yan-Dong, W. Jun-De, B. Liang, Identifying influence of nodes in complex networks with coreness centrality: Decreasing the impact of densely local connection, Chinese Physics Letters 33 (2016) 028901.

[26] J. T. Matamalas, A. Arenas, S. Gómez, Effective approach to epidemic containment using link equations in complex networks, Science Advances 4 (2018) eaau4212.

[27] C. Castellano, R. Pastor-Satorras, Competing activation mechanisms in epidemics on networks, Scientific Reports 2 (2012) 371.

[28] C. Stegehuis, R. van der Hofstad, J. S. H. van Leeuwaarden, Epidemic spreading on complex networks with community structures, Scientific Reports 6 (2016) 29748.

[29] F. Iannelli, A. Koher, D. Brockmann, P. Hovel, I. M. Sokolov, Effective distances for epidemics spreading on complex networks, Physical Review E 95 (2017) 012313.

[30] J. Friedman, T. Hastie, R. Tibshirani, Regularization paths for generalized linear models via coordinate descent, Journal of Statistical Software 33 (2010) 1–22.

[31] J. M. Wooldridge, Econometric analysis of cross section and panel data, MIT Press, 2002.

[32] Reuters, Austria says reopening shops has not accelerated coronavirus infections, https://www.reuters.com/article/us-health-coronavirus-austria-idUSKBN22H1HP, 2020. Accessed: 2020-05-08.

Supplementary Material

This Supplementary Material presents additional results of our empirical application to Brazil.

Table A1: *Estimated share of the local population with COVID-19 in Brazil at the peak and the corresponding month and day in 2020. We report the peak date and share of infectious persons to the local population of the city with and without health policy measures (see Section 4.2 for details). This simulation uses data up to May 8, 2020. We only report estimates for cities in which the simulated infectious peak with policy is higher than 5% of the local population.*

				No Policy (Peak)		With Policy (Peak)		
Row Number	Region	State	Name	Date	Infectious (%)	Date	Infectious (%)	Reduction (%)
1	Southeast	São Paulo	Ourinhos	Oct 17	10.42	Nov 30	7.03	36.57
2	Midwest	Mato Grosso	Mirassol d'Oeste	Sep 29	10.96	Nov 06	7.49	35.10
3	South	Paraná	Santa Fé	Oct 10	9.65	Nov 29	6.33	34.82
4	South	Rio Grande do Sul	Tio Hugo	Sep 19	10.43	Oct 26	7.03	34.66
5	Southeast	São Paulo	Laranjal Paulista	Oct 08	10.46	Nov 23	7.04	33.14
6	North	Pará	Cachoeira do Arari	Sep 29	10.97	Nov 02	7.67	32.55
7	South	Rio Grande do Sul	Porto Alegre	Oct 03	9.78	Nov 22	6.45	32.12
8	South	Paraná	Fazenda Rio Grande	Oct 17	9.61	Nov 22	6.83	31.72
9	Northeast	Ceará	Acopiara	Sep 20	14.79	Oct 21	11.03	31.70
10	Southeast	São Paulo	Araçariguama	Sep 26	13.25	Oct 31	9.56	31.48
11	South	Paraná	Pato Branco	Oct 01	12.13	Nov 10	8.53	31.18
12	Northeast	Rio Grande do Norte	Nísia Floresta	Sep 08	14.59	Oct 06	10.83	31.05
13	Southeast	São Paulo	São Roque	Sep 12	14.30	Oct 13	10.56	31.04
14	South	Rio Grande do Sul	Vacaria	Oct 31	10.45	Dec 21	7.04	30.44
15	South	Santa Catarina	Balneário Arroio do Silva	Sep 27	10.18	Nov 06	6.80	30.44
16	Northeast	Bahia	Feira de Santana	Oct 13	10.13	Nov 29	6.76	30.21
17	South	Santa Catarina	Pedras Grandes	Sep 10	9.88	Oct 20	6.54	30.19
18	Northeast	Paraíba	Junco do Seridó	Sep 07	11.71	Oct 13	8.15	30.02
19	Southeast	Rio de Janeiro	Barra do Piraí	Sep 30	10.70	Nov 11	7.26	29.75
20	South	Santa Catarina	Balneário Camboriú	Sep 07	10.25	Oct 12	6.87	29.54
21	South	Rio Grande do Sul	Canoas	Oct 23	10.12	Dec 14	6.76	29.49
22	Southeast	São Paulo	Lavrinhas	Aug 31	17.11	Sep 25	13.22	29.47
23	Southeast	Minas Gerais	Belo Horizonte	Sep 25	10.59	Nov 10	7.17	28.79
24	Northeast	Piauí	Piracuruca	Sep 12	12.03	Oct 18	8.44	28.63
25	Northeast	Ceará	Alto Santo	Aug 23	16.80	Sep 16	12.91	28.07
26	Northeast	Rio Grande do Norte	Tenente Ananias	Sep 15	11.61	Oct 24	8.08	28.06
27	Midwest	Mato Grosso do Sul	Campo Grande	Sep 23	12.55	Oct 31	8.92	27.80
28	South	Paraná	Paranaguá	Sep 26	12.47	Nov 06	8.84	27.79
29	Southeast	São Paulo	Angatuba	Sep 20	14.95	Oct 23	11.13	27.76
30	South	Paraná	Araruna	Aug 09	15.44	Aug 30	11.60	27.71
31	Southeast	São Paulo	Atibaia	Sep 22	11.80	Oct 30	8.25	27.42
32	Northeast	Pernambuco	Lagoa dos Gatos	Sep 16	13.36	Oct 21	9.66	27.06
33	South	Paraná	Campo Mourão	Sep 07	12.32	Oct 14	8.72	26.93
34	Southeast	São Paulo	São José do Rio Preto	Sep 23	11.69	Nov 04	8.15	26.89
35	Southeast	São Paulo	Jacareí	Oct 05	11.12	Nov 16	7.65	26.72
36	South	Santa Catarina	Florianópolis	Aug 26	12.78	Sep 28	9.13	26.70
37	Midwest	Mato Grosso	Rondonópolis	Sep 12	12.88	Oct 17	9.22	26.66
38	Northeast	Rio Grande do Norte	Parnamirim	Aug 31	13.00	Sep 28	9.33	26.41
39	Southeast	São Paulo	Taubaté	Sep 11	15.38	Oct 09	11.55	26.39
40	Southeast	São Paulo	Barra Bonita	Sep 09	14.92	Oct 05	11.30	26.22
41	Southeast	São Paulo	Peruíbe	Sep 05	13.89	Sep 28	10.77	26.19
42	South	Paraná	Umuarama	Sep 29	15.47	Oct 25	12.27	26.17
43	Southeast	Rio de Janeiro	Paraty	Sep 19	12.63	Oct 24	9.00	26.04
44	South	Rio Grande do Sul	Novo Hamburgo	Oct 04	13.51	Nov 16	9.79	25.84
45	Southeast	São Paulo	Marília	Oct 19	11.32	Dec 06	7.85	25.77
46	Southeast	Minas Gerais	Patos de Minas	Sep 15	13.92	Oct 21	10.17	25.52
47	Southeast	São Paulo	Nazaré Paulista	Aug 20	17.97	Sep 12	14.00	25.47
48	South	Paraná	Cascavel	Sep 10	14.14	Oct 17	10.38	25.33
49	Southeast	São Paulo	São José dos Campos	Sep 05	12.83	Oct 09	9.18	25.33
50	Northeast	Bahia	Itagibá	Aug 25	12.90	Sep 25	9.25	25.27
51	Southeast	Rio de Janeiro	Nova Friburgo	Sep 03	13.96	Oct 03	10.21	24.92
52	Southeast	São Paulo	Mococa	Oct 24	10.49	Dec 15	7.04	24.88
53	Southeast	São Paulo	Araçatuba	Aug 28	14.64	Sep 24	10.84	24.85
54	Southeast	Minas Gerais	Varginha	Sep 18	13.44	Oct 25	9.74	24.80
55	Midwest	Distrito Federal	Brasília	Aug 26	13.12	Sep 30	9.45	24.79
56	South	Santa Catarina	Itapema	Aug 20	17.43	Sep 12	13.47	24.62
57	South	Rio Grande do Sul	Alvorada	Sep 22	14.16	Oct 30	10.40	24.55
58	Southeast	São Paulo	Bragança Paulista	Sep 01	13.45	Oct 04	9.75	24.32
59	Northeast	Bahia	Itapetinga	Sep 19	14.78	Oct 27	10.97	24.20
60	Northeast	Pernambuco	Catende	Sep 28	12.96	Nov 09	9.32	24.11
61	Northeast	Bahia	Uruçuca	Aug 20	13.57	Sep 19	9.87	24.05
62	South	Santa Catarina	Itajaí	Aug 29	13.50	Oct 01	9.80	24.01
63	Northeast	Bahia	Lauro de Freitas	Sep 01	13.38	Oct 04	9.70	23.89

Continued on next page

Table A1 – continued from previous page

				No Policy (Peak)		With Policy (Peak)		
Row Number	Region	State	Name	Date	Infectious (%)	Date	Infectious (%)	Reduction (%)
64	Southeast	Minas Gerais	Patrocínio	Aug 24	16.43	Sep 20	12.51	23.82
65	Southeast	Rio de Janeiro	Mangaratiba	Sep 17	12.50	Oct 26	8.91	23.79
66	Southeast	Minas Gerais	Juiz de Fora	Sep 02	13.69	Oct 05	9.98	23.64
67	Southeast	São Paulo	Eldorado	Sep 17	12.77	Oct 26	9.16	23.63
68	Northeast	Bahia	Camaçari	Sep 17	13.30	Oct 21	9.64	23.62
69	Southeast	Minas Gerais	São Sebastião do Paraíso	Oct 26	12.13	Dec 14	8.58	23.59
70	Southeast	Rio de Janeiro	São Pedro da Aldeia	Sep 03	14.94	Oct 03	11.12	23.55
71	Southeast	São Paulo	São Carlos	Sep 16	15.65	Oct 20	11.78	23.44
72	Midwest	Goiás	Goiandira	Aug 14	14.83	Sep 10	11.02	23.37
73	Southeast	São Paulo	Lençóis Paulista	Oct 17	10.91	Dec 03	7.52	23.35
74	Southeast	Minas Gerais	Uberlândia	Sep 05	14.41	Oct 10	10.64	23.11
75	South	Paraná	Guairaçá	Oct 15	9.29	Aug 25	6.71	23.01
76	Midwest	Mato Grosso	Barra do Garças	Nov 12	9.31	Jan 05	6.00	22.93
77	South	Paraná	Maringá	Aug 30	15.89	Oct 02	12.01	22.91
78	Northeast	Pernambuco	Aliança	Aug 24	15.34	Sep 20	11.50	22.88
79	Midwest	Goiás	Pires do Rio	Sep 01	16.06	Oct 01	12.16	22.86
80	Southeast	São Paulo	Mineiros do Tietê	Aug 19	15.97	Sep 13	12.08	22.82
81	Southeast	São Paulo	Ferraz de Vasconcelos	Aug 20	14.72	Sep 16	10.92	22.81
82	Southeast	Rio de Janeiro	Bom Jardim	Aug 30	14.24	Sep 29	10.49	22.80
83	Southeast	São Paulo	Leme	Oct 01	14.99	Nov 09	11.17	22.79
84	Northeast	Rio Grande do Norte	Açu	Aug 09	16.04	Sep 01	12.15	22.70
85	Northeast	Rio Grande do Norte	São Gonçalo do Amarante	Aug 13	16.13	Sep 08	12.23	22.50
86	South	Rio Grande do Sul	São Leopoldo	Aug 30	14.77	Sep 30	10.97	22.50
87	Southeast	Minas Gerais	Pouso Alegre	Aug 31	14.81	Oct 02	11.01	22.48
88	Northeast	Pernambuco	Belo Jardim	Aug 31	17.17	Sep 26	13.21	22.41
89	Northeast	Pernambuco	Carnaíba	Aug 28	16.53	Sep 25	12.61	22.36
90	Southeast	São Paulo	Monte Alto	Aug 22	17.17	Sep 17	13.20	22.29
91	South	Santa Catarina	Sombrio	Aug 11	15.00	Sep 06	11.19	22.27
92	Southeast	São Paulo	Cravinhos	Aug 24	16.85	Sep 21	12.91	22.25
93	Southeast	Minas Gerais	Uberaba	Aug 25	16.58	Sep 21	12.66	22.24
94	South	Santa Catarina	Criciúma	Aug 16	14.66	Sep 15	10.87	22.22
95	Southeast	São Paulo	Presidente Venceslau	Aug 20	16.58	Sep 17	12.66	22.06
96	Southeast	Minas Gerais	Ouro Fino	Sep 08	14.71	Oct 13	10.93	21.98
97	Northeast	Sergipe	Itabaianinha	Aug 31	15.77	Sep 29	11.90	21.83
98	Southeast	Minas Gerais	Belmiro Braga	Aug 29	12.74	Sep 30	9.18	21.79
99	Southeast	São Paulo	Vinhedo	Aug 08	18.61	Aug 28	14.58	21.74
100	Northeast	Bahia	Gongogi	Aug 02	18.52	Aug 23	14.50	21.74
101	Northeast	Ceará	Aquiraz	Sep 19	11.44	Aug 17	8.69	21.65
102	Southeast	São Paulo	Vargem Grande Paulista	Aug 20	16.77	Sep 19	12.84	21.60
103	Southeast	Minas Gerais	Divinópolis	Aug 16	16.73	Sep 12	12.79	21.58
104	Southeast	São Paulo	Jaboticabal	Aug 16	17.30	Sep 12	13.33	21.56
105	Southeast	São Paulo	Capão Bonito	Aug 17	22.13	Sep 05	18.05	21.56
106	South	Santa Catarina	São Ludgero	Aug 02	15.58	Aug 27	11.73	21.48
107	Midwest	Goiás	Valparaíso de Goiás	Sep 13	14.62	Oct 17	10.85	21.41
108	Southeast	Minas Gerais	Extrema	Aug 09	15.33	Sep 01	11.51	21.40
109	Southeast	São Paulo	Caieiras	Aug 06	16.60	Aug 31	12.67	21.35
110	Southeast	São Paulo	São Caetano do Sul	Jul 26	16.58	Aug 19	12.66	21.17
111	Southeast	São Paulo	Ribeirão Pires	Aug 04	17.85	Aug 24	13.85	21.02
112	Northeast	Pernambuco	Machados	Aug 12	17.94	Sep 04	13.94	21.00
113	Southeast	Rio de Janeiro	Miguel Pereira	Aug 03	16.94	Aug 31	12.99	20.94
114	Southeast	São Paulo	Tatuí	Sep 03	16.11	Sep 28	12.44	20.86
115	Southeast	São Paulo	Rio Claro	Aug 26	18.81	Sep 23	14.77	20.84
116	Southeast	São Paulo	Assis	Sep 02	16.11	Oct 03	12.23	20.75
117	Northeast	Pernambuco	Salgueiro	Aug 20	17.41	Sep 10	13.65	20.69
118	Southeast	Rio de Janeiro	Araruama	Aug 11	18.87	Sep 03	14.83	20.65
119	Northeast	Pernambuco	Chã de Alegria	Jul 26	20.66	Aug 10	16.59	20.60
120	Southeast	Minas Gerais	Poços de Caldas	Aug 30	16.90	Sep 27	12.96	20.49
121	Northeast	Maranhão	Raposa	Aug 14	17.14	Sep 06	13.19	20.40
122	Southeast	Rio de Janeiro	Bom Jesus do Itabapoana	Aug 10	17.86	Sep 03	13.86	20.39
123	Southeast	Rio de Janeiro	Resende	Aug 17	18.32	Sep 10	14.30	20.24
124	North	Acre	Plácido de Castro	Jul 29	16.18	Aug 19	12.30	20.22
125	South	Rio Grande do Sul	Serafina Corrêa	Aug 13	14.84	Sep 09	11.07	20.14
126	Southeast	São Paulo	Cotia	Jul 29	18.26	Aug 22	14.24	19.91
127	South	Santa Catarina	Cocal do Sul	Aug 18	19.55	Sep 09	15.48	19.89
128	Northeast	Rio Grande do Norte	Mossoró	Aug 01	18.03	Aug 26	14.03	19.87
129	South	Santa Catarina	Balneário Gaivota	Aug 06	17.14	Aug 31	13.19	19.84
130	South	Santa Catarina	Urussanga	Aug 13	16.24	Sep 08	12.36	19.79
131	Northeast	Pernambuco	Frei Miguelinho	Sep 07	12.72	Oct 10	9.18	19.72
132	Southeast	Minas Gerais	Novo Cruzeiro	Aug 15	22.42	Sep 03	18.26	19.69
133	Northeast	Sergipe	Simão Dias	Aug 11	19.56	Sep 01	15.48	19.60
134	Southeast	São Paulo	Miracatu	Aug 13	18.82	Sep 03	14.78	19.56
135	Southeast	Rio de Janeiro	Barra Mansa	Aug 18	17.47	Sep 12	13.51	19.55
136	Northeast	Maranhão	São José de Ribamar	Jul 25	18.87	Aug 13	14.83	19.55

Continued on next page

Table A1 – continued from previous page

				No Policy (Peak)		With Policy (Peak)		
Row Number	Region	State	Name	Date	Infectious (%)	Date	Infectious (%)	Reduction (%)
137	Northeast	Bahia	Ipiaú	Sep 17	11.60	Nov 04	8.27	19.54
138	Midwest	Goiás	Paraúna	Jul 30	20.78	Aug 18	16.67	19.45
139	Southeast	São Paulo	Cruzeiro	Sep 01	16.77	Oct 01	12.85	19.44
140	Northeast	Pernambuco	Itapetim	Sep 20	14.96	Sep 06	11.56	19.43
141	Southeast	Minas Gerais	São Romão	Aug 01	22.90	Aug 16	18.97	19.36
142	Northeast	Paraíba	Marizópolis	Jul 16	28.95	Jul 28	24.77	19.34
143	South	Santa Catarina	Palhoça	Aug 14	20.13	Sep 08	16.03	19.34
144	Northeast	Pernambuco	Lagoa de Itaenga	Aug 14	19.96	Sep 04	15.88	19.30
145	Southeast	São Paulo	São Manuel	Aug 10	17.89	Sep 03	13.91	19.27
146	South	Santa Catarina	Gaspar	Aug 06	18.11	Aug 30	14.11	19.25
147	Southeast	São Paulo	Águas de Lindóia	Aug 05	19.40	Aug 28	15.33	19.21
148	Midwest	Goiás	Goiânia	Aug 06	19.51	Aug 29	15.43	19.09
149	Southeast	Minas Gerais	Itabira	Aug 15	21.14	Sep 07	17.00	19.00
150	North	Pará	Vigia	Aug 13	16.46	Sep 06	12.59	18.84
151	Northeast	Ceará	Russas	Aug 22	16.07	Sep 17	12.23	18.83
152	Southeast	São Paulo	Botucatu	Aug 04	19.37	Aug 26	15.31	18.81
153	Midwest	Mato Grosso	Lucas do Rio Verde	Sep 06	16.18	Oct 11	12.34	18.81
154	Southeast	Minas Gerais	Mariana	Jul 31	20.33	Aug 22	16.23	18.66
155	Northeast	Ceará	Tianguá	Aug 24	17.94	Sep 22	13.95	18.62
156	Southeast	Espírito Santo	Linhares	Aug 01	20.61	Aug 22	16.50	18.61
157	Northeast	Ceará	Ipueiras	Aug 13	19.66	Sep 05	15.58	18.59
158	Midwest	Goiás	Luziânia	Aug 13	20.85	Sep 09	16.72	18.58
159	Southeast	Espírito Santo	São Mateus	Aug 10	18.98	Sep 04	14.94	18.55
160	Northeast	Ceará	Sobral	Jul 27	19.92	Aug 13	15.84	18.44
161	Northeast	Ceará	Jaguaribe	Jul 31	21.01	Aug 22	16.88	18.43
162	Southeast	São Paulo	Mairiporã	Aug 05	19.35	Aug 29	15.29	18.41
163	Southeast	São Paulo	Santo André	Jul 23	20.00	Aug 13	15.91	18.33
164	Midwest	Mato Grosso	Cáceres	Aug 15	19.85	Sep 08	15.77	18.23
165	Southeast	São Paulo	Santos	Jul 15	20.55	Aug 03	16.44	18.23
166	Northeast	Alagoas	Murici	Jul 27	19.40	Aug 10	16.02	18.21
167	South	Paraná	Guaíra	Jul 21	23.31	Aug 07	19.11	18.09
168	Southeast	São Paulo	Monte Mor	Nov 04	10.63	Aug 13	8.54	18.07
169	Northeast	Ceará	Santa Quitéria	Aug 04	20.02	Aug 28	15.93	18.05
170	Southeast	Espírito Santo	Presidente Kennedy	Jul 19	22.25	Aug 03	18.08	18.03
171	Southeast	São Paulo	Mirandópolis	Aug 24	16.44	Sep 20	12.61	17.96
172	Northeast	Rio Grande do Norte	Natal	Jul 23	20.83	Aug 12	16.70	17.89
173	Southeast	Rio de Janeiro	Arraial do Cabo	Jul 28	20.80	Aug 19	16.68	17.88
174	Northeast	Pernambuco	Palmares	Aug 14	19.91	Sep 08	15.83	17.85
175	Southeast	São Paulo	Itaquaquecetuba	Jul 25	23.11	Aug 10	18.93	17.80
176	Southeast	São Paulo	Mauá	Jul 26	21.31	Aug 13	17.17	17.78
177	North	Pará	São João do Araguaia	Nov 13	10.15	Jul 20	8.28	17.58
178	Southeast	Rio de Janeiro	Teresópolis	Jul 30	21.75	Aug 17	17.60	17.58
179	Northeast	Alagoas	Matriz de Camaragibe	Nov 28	8.86	Jul 13	7.24	17.57
180	Southeast	Rio de Janeiro	Iguaba Grande	Jul 17	21.89	Aug 04	17.73	17.54
181	Southeast	São Paulo	Indaiatuba	Aug 03	23.92	Aug 20	19.72	17.52
182	Southeast	São Paulo	Paulínia	Aug 06	21.31	Aug 24	17.17	17.51
183	Southeast	Espírito Santo	Fundão	Jul 17	21.28	Jul 28	17.84	17.48
184	Southeast	Rio de Janeiro	Tanguá	Jul 24	22.62	Aug 11	18.44	17.45
185	Southeast	Rio de Janeiro	Rio de Janeiro	Jul 17	21.50	Aug 07	17.35	17.44
186	Northeast	Ceará	São Luís do Curu	Aug 15	16.19	Sep 10	12.41	17.41
187	Northeast	Rio Grande do Norte	Canguaretama	Jul 26	21.25	Aug 13	17.12	17.40
188	Northeast	Piauí	Pedro II	Aug 09	22.84	Aug 29	18.66	17.39
189	Southeast	São Paulo	Santana de Parnaíba	Jul 17	21.57	Aug 01	17.43	17.39
190	South	Santa Catarina	Concórdia	Jul 20	20.82	Jul 31	17.39	17.37
191	Southeast	São Paulo	Valinhos	Aug 01	22.04	Aug 21	17.88	17.37
192	Northeast	Pernambuco	Caruaru	Aug 12	21.24	Sep 04	17.10	17.33
193	Southeast	Rio de Janeiro	Maricá	Jul 24	21.85	Aug 12	17.70	17.32
194	Southeast	São Paulo	Mogi Guaçu	Aug 08	22.24	Aug 30	18.07	17.32
195	Southeast	São Paulo	Avaré	Aug 06	21.66	Aug 27	17.51	17.29
196	Northeast	Maranhão	Zé Doca	Aug 15	18.99	Sep 06	14.97	17.23
197	Northeast	Ceará	Crateús	Aug 12	19.08	Sep 02	15.06	17.23
198	South	Santa Catarina	Joinville	Jul 24	22.19	Aug 13	18.03	17.22
199	Southeast	São Paulo	Embu das Artes	Aug 02	21.86	Aug 25	17.70	17.17
200	Southeast	Rio de Janeiro	Niterói	Jul 18	22.20	Aug 09	18.03	17.07
201	Northeast	Ceará	Ocara	Aug 17	16.51	Sep 11	12.71	17.07
202	Southeast	São Paulo	Araraquara	Jul 28	23.39	Aug 15	19.19	16.99
203	Southeast	Rio de Janeiro	Nilópolis	Jul 21	22.72	Aug 05	18.54	16.94
204	Southeast	São Paulo	São Bernardo do Campo	Jul 17	22.39	Aug 06	18.22	16.91
205	Southeast	São Paulo	Votorantim	Aug 06	22.26	Aug 26	18.09	16.91
206	Northeast	Pernambuco	Abreu e Lima	Jul 13	24.35	Jul 27	20.14	16.89
207	Southeast	São Paulo	Catanduva	Aug 13	22.05	Sep 06	17.89	16.85
208	Midwest	Mato Grosso do Sul	Três Lagoas	Jul 20	24.26	Aug 04	20.04	16.78
209	South	Santa Catarina	Braço do Norte	Jul 01	22.61	Jul 14	18.43	16.64

Continued on next page

Table A1 – continued from previous page

				No Policy (Peak)		With Policy (Peak)		
Row Number	Region	State	Name	Date	Infectious (%)	Date	Infectious (%)	Reduction (%)
210	Northeast	Pernambuco	Araçoiaba	Jul 18	25.23	Jul 31	21.02	16.54
211	Southeast	Rio de Janeiro	Cabo Frio	Jul 22	25.16	Aug 06	20.93	16.47
212	Southeast	Rio de Janeiro	Volta Redonda	Jul 12	23.30	Jul 31	19.11	16.45
213	Southeast	São Paulo	Itanhaém	Jul 23	25.53	Aug 08	21.29	16.43
214	Southeast	Espírito Santo	Guarapari	Jul 21	25.70	Aug 03	21.71	16.42
215	Southeast	Rio de Janeiro	Magé	Jul 21	23.12	Aug 07	18.92	16.40
216	Northeast	Pernambuco	Macaparana	Jul 19	24.02	Aug 06	19.81	16.28
217	Northeast	Pernambuco	Arcoverde	Jul 26	22.97	Aug 14	18.78	16.27
218	North	Pará	Óbidos	Aug 03	24.36	Aug 21	20.14	16.22
219	Southeast	Rio de Janeiro	Queimados	Jul 18	24.11	Aug 03	19.89	16.20
220	Southeast	Espírito Santo	Aracruz	Jul 22	25.14	Aug 06	20.91	16.20
221	North	Pará	Igarapé-Açu	Jul 18	25.25	Jul 28	21.73	16.18
222	Northeast	Pernambuco	Ribeirão	Jul 25	26.77	Aug 08	22.56	16.16
223	Northeast	Bahia	Salvador	Jul 13	23.39	Aug 01	19.19	16.13
224	Southeast	São Paulo	Nova Odessa	Jul 31	22.32	Aug 22	18.16	16.10
225	Northeast	Rio Grande do Norte	Encanto	Jul 04	25.53	Jul 16	21.29	15.99
226	Southeast	Espírito Santo	Vitória	Jul 05	24.08	Jul 20	19.87	15.99
227	Southeast	São Paulo	Diadema	Jul 13	23.97	Jul 31	19.75	15.95
228	Southeast	Minas Gerais	Governador Valadares	Aug 01	24.07	Aug 21	19.85	15.94
229	Northeast	Maranhão	Davinópolis	Aug 08	20.75	Aug 29	16.72	15.94
230	Northeast	Ceará	Fortaleza	Jun 30	23.92	Jul 16	19.71	15.89
231	Northeast	Ceará	Iguatu	Jul 23	24.35	Aug 10	20.13	15.86
232	Southeast	São Paulo	Itapetininga	Aug 02	23.22	Aug 22	19.03	15.85
233	Northeast	Alagoas	São Miguel dos Milagres	Aug 08	18.78	Aug 31	14.79	15.84
234	Southeast	Rio de Janeiro	Rio Bonito	Jul 12	24.15	Jul 30	19.94	15.83
235	Northeast	Pernambuco	Ipojuca	Jul 19	24.52	Aug 04	20.30	15.80
236	Southeast	São Paulo	Franca	Aug 09	24.26	Aug 29	20.04	15.74
237	Southeast	São Paulo	Arujá	Jul 12	24.18	Jul 29	19.96	15.74
238	Southeast	São Paulo	Itapecerica da Serra	Jul 15	24.50	Aug 01	20.27	15.72
239	Northeast	Pernambuco	Panelas	Aug 01	25.63	Aug 17	21.40	15.65
240	Southeast	Rio de Janeiro	Macaé	Jul 19	26.10	Aug 04	21.85	15.63
241	Southeast	Rio de Janeiro	Nova Iguaçu	Jul 16	25.32	Aug 03	21.09	15.61
242	Southeast	São Paulo	Campinas	Jul 20	24.86	Aug 08	20.63	15.58
243	North	Pará	Santa Cruz do Arari	Jul 21	21.63	Aug 05	17.52	15.56
244	South	Santa Catarina	Indaial	Jul 22	24.37	Aug 08	20.15	15.56
245	Northeast	Bahia	Ilhéus	Jul 04	25.29	Jul 18	21.05	15.55
246	Southeast	São Paulo	Dracena	Jul 23	23.75	Aug 11	19.54	15.53
247	Northeast	Pernambuco	Limoeiro	Jul 20	25.51	Aug 05	21.27	15.52
248	Southeast	São Paulo	Poá	Jul 12	25.12	Jul 29	20.89	15.51
249	North	Amazonas	Manaus	Jul 03	24.79	Jul 20	20.56	15.50
250	Northeast	Ceará	Amontada	Sep 18	16.62	Sep 04	13.32	15.50
251	Southeast	São Paulo	Mogi das Cruzes	Jul 13	24.83	Jul 30	20.60	15.50
252	Southeast	Minas Gerais	Araxá	Nov 04	12.40	Jul 28	10.28	15.48
253	Southeast	São Paulo	Francisco Morato	Jul 20	24.64	Aug 07	20.41	15.47
254	Southeast	São Paulo	Várzea Paulista	Jul 28	25.13	Aug 13	20.89	15.46
255	Southeast	São Paulo	Ribeirão Preto	Aug 03	25.12	Aug 26	20.89	15.43
256	North	Amazonas	Itapiranga	Jul 01	27.65	Jul 09	24.12	15.41
257	Southeast	São Paulo	Sumaré	Sep 29	15.18	Aug 17	12.65	15.40
258	Southeast	São Paulo	Itu	Jul 20	28.21	Aug 03	23.95	15.39
259	Southeast	Rio de Janeiro	Petrópolis	Jul 10	25.49	Jul 28	21.25	15.39
260	Northeast	Ceará	Limoeiro do Norte	Jul 19	24.45	Aug 05	20.24	15.38
261	North	Tocantins	Palmas	Jul 26	23.06	Aug 13	18.88	15.34
262	Southeast	São Paulo	Barueri	Jul 03	24.90	Jul 18	20.67	15.33
263	North	Pará	Marituba	Jul 14	26.54	Jul 28	22.28	15.32
264	Northeast	Pernambuco	Bom Jardim	Jul 17	25.93	Aug 03	21.68	15.28
265	Northeast	Pernambuco	Bonito	Jul 23	27.46	Aug 06	23.19	15.28
266	Northeast	Pernambuco	Garanhuns	Jul 30	25.83	Aug 18	21.58	15.20
267	Northeast	Pernambuco	São Lourenço da Mata	Jul 07	26.14	Jul 22	21.89	15.17
268	South	Paraná	Paranavaí	Jul 18	24.02	Aug 03	19.81	15.12
269	Southeast	Espírito Santo	Vila Velha	Jun 30	25.87	Jul 15	21.62	15.06
270	North	Roraima	Alto Alegre	Jul 17	24.05	Aug 03	19.84	15.04
271	North	Amazonas	Manicoré	Jul 28	24.17	Aug 16	19.96	15.04
272	North	Pará	Curionópolis	Sep 13	16.94	Aug 29	13.49	15.03
273	Northeast	Piauí	Buriti dos Lopes	Sep 04	20.18	Aug 17	16.61	15.02
274	North	Pará	Santarém	Jul 31	25.23	Aug 22	20.99	15.00
275	Southeast	São Paulo	São Paulo	Jul 02	25.84	Jul 20	21.60	15.00
276	Southeast	São Paulo	Santa Bárbara d'Oeste	Aug 07	25.38	Aug 25	21.14	14.96
277	North	Amazonas	Novo Airão	Jul 27	21.65	Aug 17	17.54	14.88
278	Northeast	Pernambuco	Paudalho	Jul 11	26.22	Jul 27	21.97	14.78
279	Northeast	Pernambuco	Lagoa do Carro	Jul 17	23.35	Aug 05	19.17	14.76
280	Northeast	Rio Grande do Norte	Apodi	Jul 14	24.46	Jul 30	20.25	14.75
281	Southeast	Minas Gerais	Santos Dumont	Aug 09	21.85	Aug 29	17.72	14.74
282	Northeast	Ceará	Pacatuba	Jul 19	23.00	Aug 03	18.83	14.71

Continued on next page

Table A1 – continued from previous page

				No Policy (Peak)		With Policy (Peak)		
Row Number	Region	State	Name	Date	Infectious (%)	Date	Infectious (%)	Reduction (%)
283	Southeast	São Paulo	Presidente Prudente	Jul 24	25.90	Aug 11	21.65	14.65
284	Southeast	São Paulo	Mongaguá	Jul 15	28.05	Jul 29	23.78	14.64
285	Northeast	Ceará	Horizonte	Jul 08	25.44	Jul 22	21.20	14.64
286	Southeast	São Paulo	Guarujá	Jul 09	28.02	Jul 22	23.75	14.60
287	North	Roraima	Boa Vista	Jun 30	26.40	Jul 15	22.14	14.58
288	Southeast	São Paulo	Taboão da Serra	Jul 16	27.36	Aug 03	23.09	14.55
289	Northeast	Ceará	Capistrano	Jul 19	24.87	Aug 05	20.64	14.54
290	Southeast	Rio de Janeiro	Rio das Ostras	Jul 09	26.91	Jul 25	22.65	14.52
291	Southeast	São Paulo	Araras	Jul 30	24.63	Aug 14	20.65	14.46
292	Southeast	São Paulo	Itapevi	Jul 09	26.83	Jul 24	22.57	14.46
293	Northeast	Bahia	Itabuna	Jul 03	28.64	Jul 15	24.36	14.40
294	Northeast	Maranhão	Paço do Lumiar	Jul 03	27.30	Jul 17	23.03	14.36
295	Northeast	Pernambuco	Paulista	Jun 29	27.43	Jul 11	23.16	14.34
296	Northeast	Ceará	Maracanaú	Jul 03	27.05	Jul 17	22.78	14.34
297	Northeast	Ceará	Massapê	Aug 19	17.96	Sep 14	14.11	14.32
298	Northeast	Piauí	Canto do Buriti	Aug 15	22.75	Sep 06	18.93	14.29
299	Southeast	São Paulo	Praia Grande	Jul 08	27.67	Jul 23	23.40	14.26
300	Southeast	São Paulo	Limeira	Jul 20	27.26	Aug 05	23.00	14.24
301	Southeast	São Paulo	Franco da Rocha	Jul 02	27.80	Jul 17	23.53	14.22
302	Southeast	São Paulo	Campo Limpo Paulista	Jul 11	28.92	Jul 25	24.65	14.18
303	Southeast	São Paulo	Embu-Guaçu	Jul 07	29.87	Jul 18	25.59	14.15
304	South	Rio Grande do Sul	Marau	Jun 25	27.43	Jul 07	23.16	14.15
305	Southeast	Rio de Janeiro	Belford Roxo	Jul 16	28.10	Aug 02	23.83	14.15
306	Southeast	Rio de Janeiro	Guapimirim	Jul 17	28.81	Jul 29	24.79	14.14
307	Southeast	São Paulo	São Lourenço da Serra	Jul 07	30.04	Jul 19	25.77	14.13
308	North	Acre	Rio Branco	Jul 02	26.11	Jul 17	21.86	14.10
309	Southeast	São Paulo	Agudos	Jul 07	28.88	Jul 21	24.60	14.10
310	Northeast	Alagoas	Maragogi	Jul 15	28.37	Jul 25	24.82	14.10
311	Southeast	Espírito Santo	Cariacica	Jul 05	27.50	Jul 19	23.23	14.09
312	Southeast	Rio de Janeiro	Duque de Caxias	Jul 10	28.26	Jul 26	23.98	14.03
313	South	Rio Grande do Sul	Garibaldi	Jul 04	29.01	Jul 14	24.74	14.03
314	South	Santa Catarina	Penha	Jul 19	27.59	Aug 03	23.32	14.01
315	Southeast	Rio de Janeiro	Itaboraí	Jul 03	27.09	Jul 17	22.83	13.98
316	Southeast	Rio de Janeiro	Paracambi	Jul 10	27.46	Jul 22	23.19	13.95
317	Northeast	Sergipe	Aracaju	Jul 12	25.35	Jul 28	21.12	13.95
318	Southeast	São Paulo	Suzano	Jul 08	28.35	Jul 23	24.07	13.95
319	Southeast	Rio de Janeiro	São Gonçalo	Jul 11	28.09	Jul 27	23.82	13.94
320	Southeast	São Paulo	Pindamonhangaba	Jul 20	28.63	Aug 04	24.35	13.89
321	Northeast	Rio Grande do Norte	Ipanguaçu	Jul 01	32.44	Jul 11	28.18	13.85
322	North	Pará	Castanhal	Jul 07	28.51	Jul 18	24.23	13.83
323	North	Pará	Marapanim	Jul 15	26.37	Jul 25	22.83	13.82
324	Southeast	São Paulo	Piracicaba	Jul 14	27.43	Jul 29	23.16	13.82
325	Southeast	Espírito Santo	Serra	Jun 28	28.66	Jul 11	24.38	13.82
326	Northeast	Pernambuco	Tabira	Jul 31	25.28	Aug 17	21.07	13.80
327	Northeast	Pernambuco	Moreno	Jul 03	29.51	Jul 15	25.22	13.77
328	Southeast	Rio de Janeiro	Cachoeiras de Macacu	Jul 11	28.30	Jul 25	24.02	13.75
329	Southeast	São Paulo	Piracaia	Jul 13	30.11	Jul 25	25.83	13.69
330	Southeast	São Paulo	Carapicuíba	Jul 02	28.38	Jul 17	24.10	13.69
331	Southeast	São Paulo	Itatiba	Jul 18	28.04	Aug 04	23.77	13.69
332	North	Pará	Parauapebas	Jul 06	29.87	Jul 18	25.59	13.61
333	Northeast	Pernambuco	Glória do Goitá	Jul 04	31.22	Jul 15	26.94	13.60
334	Northeast	Paraíba	Cabedelo	Jul 10	27.49	Jul 26	23.22	13.59
335	Northeast	Paraíba	Patos	Jul 12	27.61	Jul 29	23.34	13.59
336	Southeast	São Paulo	Jandira	Jul 08	28.37	Jul 22	24.09	13.57
337	North	Pará	Canaã dos Carajás	Jul 15	28.22	Jul 30	23.94	13.56
338	South	Rio Grande do Sul	Passo Fundo	Jun 29	28.71	Jul 12	24.44	13.54
339	Southeast	São Paulo	Sorocaba	Jul 13	28.91	Jul 29	24.62	13.52
340	Southeast	São Paulo	Registro	Jul 25	27.18	Aug 10	22.92	13.51
341	Southeast	São Paulo	Osasco	Jul 08	28.88	Jul 25	24.60	13.50
342	Southeast	Minas Gerais	Nova Serrana	Aug 06	25.82	Aug 25	21.59	13.50
343	North	Amapá	Macapá	Jun 27	28.81	Jul 11	24.52	13.50
344	Northeast	Pernambuco	Cabo de Santo Agostinho	Jul 12	28.30	Jul 28	24.03	13.49
345	North	Pará	Santo Antônio do Tauá	Jun 28	29.17	Jul 07	24.89	13.45
346	Northeast	Pernambuco	Carpina	Jul 08	33.15	Jul 18	28.89	13.43
347	Northeast	Rio Grande do Norte	São José de Mipibu	Jul 02	32.22	Jul 11	27.95	13.32
348	Northeast	Pernambuco	Nazaré da Mata	Jul 08	31.08	Jul 18	27.05	13.30
349	Southeast	Rio de Janeiro	São João de Meriti	Jul 10	30.06	Jul 25	25.77	13.25
350	North	Amazonas	Manacapuru	Jun 19	29.70	Jul 01	25.42	13.21
351	Northeast	Piauí	Teresina	Jun 30	28.95	Jul 15	24.68	13.16
352	Southeast	São Paulo	Guarulhos	Jul 06	29.89	Jul 22	25.60	13.13
353	Southeast	São Paulo	Jundiaí	Jul 04	29.90	Jul 18	25.61	13.12
354	Southeast	São Paulo	Americana	Jul 04	30.70	Jul 18	26.41	13.12
355	Northeast	Paraíba	João Pessoa	Jun 30	29.48	Jul 13	25.20	13.10

Continued on next page

Table A1 – continued from previous page

				No Policy (Peak)		With Policy (Peak)		
Row Number	Region	State	Name	Date	Infectious (%)	Date	Infectious (%)	Reduction (%)
356	Northeast	Pernambuco	Timbaúba	Jul 11	30.69	Jul 24	26.41	13.10
357	North	Rondônia	Ji-Paraná	Jul 03	31.99	Jul 14	27.70	13.07
358	Northeast	Pernambuco	Camaragibe	Jun 26	30.48	Jul 09	26.19	13.06
359	Southeast	São Paulo	Bauru	Jul 09	30.51	Jul 23	26.22	13.06
360	North	Pará	Santa Izabel do Pará	Jun 25	32.94	Jul 04	28.67	13.03
361	Southeast	São Paulo	São Sebastião	Jun 30	28.18	Jul 15	23.91	13.03
362	Northeast	Paraíba	Campina Grande	Jul 11	31.34	Jul 23	27.06	12.99
363	Southeast	Espírito Santo	Santa Maria de Jetibá	Jul 10	30.19	Jul 22	25.91	12.99
364	Southeast	Rio de Janeiro	Japeri	Jul 04	31.67	Jul 16	27.39	12.99
365	Northeast	Pernambuco	Goiana	Jul 03	29.37	Jul 17	25.09	12.94
366	Southeast	São Paulo	Barretos	Jul 03	31.74	Jul 14	27.46	12.93
367	North	Amapá	Oiapoque	Jun 30	30.19	Jul 10	25.93	12.91
368	Southeast	Rio de Janeiro	Mesquita	Jul 08	31.20	Jul 23	26.91	12.90
369	North	Amazonas	Itacoatiara	Jun 24	32.06	Jul 04	27.78	12.89
370	South	Rio Grande do Sul	Lajeado	Jun 24	31.45	Jul 03	27.16	12.84
371	Southeast	Rio de Janeiro	Itaguaí	Jul 03	30.80	Jul 15	26.51	12.83
372	Southeast	Minas Gerais	Montes Claros	Jul 18	30.59	Aug 03	26.30	12.78
373	Northeast	Rio Grande do Norte	Alexandria	Jun 26	37.70	Jul 04	33.51	12.73
374	Southeast	Rio de Janeiro	Sapucaia	Jun 26	30.32	Jul 07	26.03	12.73
375	Northeast	Ceará	Quixeramobim	Jul 03	29.12	Jul 17	24.84	12.72
376	Northeast	Paraíba	Taperoá	Aug 01	22.21	Aug 23	18.16	12.72
377	North	Pará	Ananindeua	Jun 29	30.87	Jul 10	26.58	12.70
378	Southeast	São Paulo	Hortolândia	Jul 20	27.43	Aug 03	23.20	12.70
379	Northeast	Sergipe	Itaporanga d'Ajuda	Jul 21	27.16	Aug 05	22.91	12.69
380	Northeast	Pernambuco	Sertânia	Jul 06	30.20	Jul 19	25.91	12.69
381	Midwest	Goiás	Aparecida de Goiânia	Jul 13	31.02	Jul 27	26.73	12.61
382	Southeast	São Paulo	Cajamar	Jun 29	30.01	Jul 11	25.72	12.60
383	North	Pará	Belém	Jun 24	31.08	Jul 06	26.80	12.55
384	Northeast	Ceará	Bela Cruz	Jul 14	26.71	Jul 25	22.71	12.52
385	Northeast	Paraíba	Santa Rita	Jun 30	30.23	Jul 12	25.94	12.51
386	South	Rio Grande do Sul	Venâncio Aires	Jul 03	35.03	Jul 13	30.78	12.50
387	North	Pará	Paragominas	Jul 01	35.46	Jul 10	31.23	12.49
388	Northeast	Pernambuco	Amaraji	Jul 04	31.59	Jul 15	27.30	12.49
389	Northeast	Maranhão	Bacabal	Jul 01	34.43	Jul 11	30.18	12.48
390	North	Amapá	Santana	Jun 30	30.79	Jul 13	26.50	12.47
391	Southeast	São Paulo	São Vicente	Jun 30	32.44	Jul 13	28.16	12.41
392	Southeast	Rio de Janeiro	São Francisco de Itabapoana	Jul 04	31.61	Jul 16	27.32	12.32
393	Southeast	São Paulo	Lins	Jul 04	34.21	Jul 16	29.95	12.31
394	Southeast	São Paulo	Juquitiba	Jul 05	32.36	Jul 17	28.07	12.27
395	North	Amazonas	São Paulo de Olivença	Jun 25	30.06	Jul 07	25.77	12.23
396	Northeast	Pernambuco	Jaboatão dos Guararapes	Jun 27	32.38	Jul 09	28.10	12.21
397	Northeast	Maranhão	São Luís	Jun 19	32.59	Jul 01	28.31	12.09
398	Northeast	Rio Grande do Norte	São Rafael	Jun 23	33.99	Jul 02	29.70	12.09
399	Southeast	São Paulo	Salesópolis	Jul 16	28.92	Jul 29	24.65	12.07
400	Northeast	Paraíba	Cajazeiras	Jul 07	33.28	Jul 18	29.00	11.92
401	Northeast	Ceará	Caucaia	Jun 23	32.69	Jul 04	28.41	11.90
402	Southeast	São Paulo	Santa Isabel	Jul 05	33.08	Jul 16	28.80	11.89
403	Northeast	Alagoas	Paripueira	Sep 21	18.12	Jul 24	15.24	11.84
404	Southeast	Espírito Santo	Viana	Jun 25	32.14	Jul 06	27.86	11.83
405	Southeast	São Paulo	Caraguatatuba	Jun 27	33.21	Jul 10	28.93	11.82
406	Northeast	Pernambuco	Itapissuma	Jul 16	26.95	Jul 29	22.96	11.78
407	Southeast	São Paulo	Lucélia	Jul 08	30.50	Jul 19	26.21	11.75
408	Northeast	Pernambuco	Vitória de Santo Antão	Jun 24	34.38	Jul 05	30.12	11.73
409	Southeast	Minas Gerais	Três Pontas	Jul 02	35.96	Jul 12	31.71	11.71
410	North	Pará	Capanema	Jun 27	33.88	Jul 05	29.63	11.70
411	North	Amazonas	Iranduba	Jun 13	34.15	Jun 21	29.88	11.65
412	North	Amazonas	Parintins	Jun 16	32.17	Jun 28	27.88	11.63
413	North	Pará	São Miguel do Guamá	Jun 28	33.16	Jul 07	28.87	11.58
414	Northeast	Piauí	Parnaíba	Jun 27	32.56	Jul 11	28.28	11.55
415	Northeast	Pernambuco	Recife	Jun 12	33.78	Jun 24	29.50	11.54
416	Northeast	Maranhão	Cururupu	Jul 03	34.12	Jul 14	29.85	11.49
417	Northeast	Ceará	Itaitinga	Jun 21	33.93	Jul 03	29.65	11.48
418	Northeast	Alagoas	Marechal Deodoro	Jun 25	34.17	Jul 05	29.89	11.48
419	North	Amazonas	Autazes	Jun 13	39.32	Jun 19	35.17	11.45
420	Northeast	Ceará	Pedra Branca	Jun 30	33.41	Jul 11	29.13	11.45
421	North	Amazonas	Borba	Jul 07	33.62	Jul 14	30.07	11.42
422	Northeast	Ceará	Maranguape	Jun 23	35.27	Jul 03	31.00	11.38
423	Northeast	Paraíba	Bayeux	Jul 02	34.23	Jul 14	29.96	11.36
424	North	Amazonas	Maués	Jun 24	35.68	Jul 04	31.42	11.35
425	Northeast	Maranhão	Imperatriz	Jun 23	34.44	Jul 03	30.17	11.22
426	North	Pará	Benevides	Jun 21	37.73	Jun 28	33.51	11.19
427	North	Rondônia	Porto Velho	Jun 19	34.43	Jul 01	30.16	11.18
428	North	Pará	Cametá	Jul 04	32.31	Jul 18	28.02	11.18

Continued on next page

Table A1 – continued from previous page

				No Policy (Peak)		With Policy (Peak)		
Row Number	Region	State	Name	Date	Infectious (%)	Date	Infectious (%)	Reduction (%)
429	Northeast	Pernambuco	Pombos	Jun 18	35.87	Jun 27	31.62	11.14
430	Northeast	Ceará	Cascavel	Jun 27	34.50	Jul 07	30.23	11.14
431	South	Paraná	São João do Caiuá	Jun 08	38.46	Jun 13	34.24	11.06
432	South	Rio Grande do Sul	Tunas	Jun 21	39.07	Jun 28	35.13	11.05
433	Northeast	Ceará	Pindoretama	Jun 25	36.54	Jul 04	32.31	11.00
434	Northeast	Paraíba	Conde	Jun 18	39.68	Jun 25	35.50	10.97
435	Northeast	Pernambuco	Itaquitinga	Jun 28	37.43	Jul 07	33.20	10.79
436	North	Amazonas	Beruri	Jun 22	39.10	Jun 30	34.91	10.76
437	North	Pará	Barcarena	Jun 26	38.44	Jul 05	34.22	10.75
438	Northeast	Pernambuco	Igarassu	Jun 21	36.99	Jul 01	32.74	10.75
439	North	Pará	Marabá	Jun 28	37.72	Jul 07	33.50	10.67
440	Southeast	São Paulo	Guararema	Jul 26	25.91	Aug 12	22.02	10.66
441	Northeast	Alagoas	Maceió	Jun 15	36.01	Jun 26	31.75	10.60
442	North	Tocantins	Paraíso do Tocantins	Jul 10	33.05	Jul 23	28.76	10.50
443	Northeast	Ceará	Eusébio	Jun 11	37.05	Jun 21	32.80	10.47
444	North	Amazonas	Barcelos	Jul 07	33.37	Jul 18	29.08	10.46
445	North	Amapá	Laranjal do Jari	Jun 17	37.99	Jun 24	33.79	10.45
446	Northeast	Pernambuco	Olinda	Jun 13	37.55	Jun 23	33.31	10.41
447	Northeast	Sergipe	Rosário do Catete	Jun 27	37.67	Jul 05	33.46	10.40
448	North	Pará	Viseu	Sep 20	21.82	Jul 22	18.90	10.38
449	Northeast	Pernambuco	Água Preta	Jun 26	38.68	Jul 04	34.50	10.37
450	North	Amazonas	Manaquiri	Jun 19	42.75	Jun 26	38.65	10.27
451	Southeast	Espírito Santo	Afonso Cláudio	Jun 18	40.63	Jun 26	36.47	10.20
452	Northeast	Pernambuco	São José da Coroa Grande	Jun 15	38.38	Jun 24	34.15	10.13
453	Northeast	Maranhão	Mirinzal	Jun 18	42.98	Jun 25	38.88	10.13
454	North	Amazonas	Coari	Jun 10	41.64	Jun 16	37.49	10.08
455	North	Pará	Bragança	Jun 22	38.53	Jun 29	34.31	10.06
456	Northeast	Maranhão	Anajatuba	Jun 23	38.13	Jul 02	33.89	9.95
457	Northeast	Pernambuco	Trindade	Jun 24	41.12	Jul 02	36.95	9.92
458	Northeast	Maranhão	Santa Rita	Jun 18	36.87	Jun 24	33.34	9.76
459	Southeast	São Paulo	Pariquera-Açu	Jun 11	41.34	Jun 17	37.17	9.74
460	North	Amazonas	Presidente Figueiredo	Jun 11	38.84	Jun 19	34.62	9.74
461	Midwest	Goiás	Planaltina	Jun 26	41.66	Jul 05	37.49	9.54
462	Northeast	Ceará	Pacajus	Jun 22	38.11	Jun 28	34.59	9.48
463	Northeast	Ceará	Trairi	Jun 26	38.38	Jul 05	34.15	9.36
464	Northeast	Ceará	Umirim	Jul 02	35.27	Jul 13	31.00	9.29
465	North	Pará	Limoeiro do Ajuru	Jun 24	38.19	Jul 03	33.96	9.29
466	Southeast	Rio de Janeiro	Campos dos Goytacazes	Jun 20	42.22	Jun 28	38.07	9.27
467	South	Paraná	Santo Antônio do Caiuá	Jun 02	44.72	Jun 06	40.64	9.16
468	Northeast	Maranhão	Arari	Jun 12	43.74	Jun 19	39.62	9.14
469	Northeast	Ceará	São Gonçalo do Amarante	Jun 11	43.73	Jun 18	39.61	8.68
470	Southeast	São Paulo	Serrana	Jun 25	39.36	Jul 04	35.13	8.67
471	North	Amazonas	Tabatinga	Jun 05	42.88	Jun 12	38.75	8.65
472	North	Pará	Ponta de Pedras	Jun 13	44.19	Jun 19	40.09	8.61
473	North	Amazonas	Careiro	Jun 06	44.39	Jun 13	40.27	8.57
474	Northeast	Pernambuco	Custódia	Jun 15	46.14	Jun 22	42.08	8.53
475	South	Rio Grande do Sul	Bento Gonçalves	Jun 11	47.34	Jun 16	43.59	8.47
476	Northeast	Ceará	Solonópole	Jun 08	49.18	Jun 13	45.26	8.30
477	Northeast	Maranhão	Lago da Pedra	Jun 18	45.01	Jun 25	40.92	8.29
478	Northeast	Paraíba	Sapé	Jun 09	45.89	Jun 16	41.84	8.25
479	Southeast	Minas Gerais	São Francisco	Jun 12	48.72	Jun 18	44.75	8.23
480	Southeast	Minas Gerais	Mário Campos	Aug 07	31.06	Jun 25	28.08	8.20
481	Northeast	Maranhão	Morros	Jun 19	42.01	Jun 24	38.55	8.18
482	Northeast	Piauí	Picos	Jun 09	46.83	Jun 15	42.82	8.17
483	North	Amazonas	Tefé	Jun 08	44.59	Jun 15	40.49	8.11
484	North	Amazonas	Urucará	Jun 07	45.59	Jun 12	41.53	7.93
485	North	Amazonas	Rio Preto da Eva	May 29	50.58	Jun 02	46.70	7.74
486	North	Pará	Breves	Jun 08	48.58	Jun 13	44.60	7.65
487	North	Pará	São Caetano de Odivelas	Jun 10	46.87	Jun 16	42.81	7.60
488	Northeast	Ceará	Acaraú	Jun 02	52.90	Jun 08	49.04	7.04
489	North	Amazonas	Carauari	May 20	56.60	May 24	52.91	6.40
490	Northeast	Ceará	Viçosa do Ceará	Jun 01	58.12	Jun 05	54.51	5.91
491	Northeast	Paraíba	Mari	May 29	62.13	Jun 01	58.73	5.57

CHAPTER

4

CHARACTERIZING DATA PATTERNS WITH CORE–PERIPHERY NETWORK MODELING

Author contribution statement

Jianglong Yan: Conceptualization, Methodology, Software, Writing original draft, Writing review and editing, Formal analysis, Validation. Leandro Anghinoni: Conceptualization, Methodology, Software, Writing original draft, Writing – review and editing, Formal analysis, Validation. Yu-Tao Zhu: Conceptualization, Methodology, Writing original draft. Weiguang Liu: Conceptualization, Methodology, Writing original draft. Gen Li: Software, Writing review and editing. Qiusheng Zheng: Conceptualization, Methodology, Writing original draft. Liang Zhao: Conceptualization, Methodology, Writing original draft, Writing review and editing, Formal analysis, Validation.

Journal of Computational Science 66 (2023) 101912

Contents lists available at ScienceDirect

Journal of Computational Science

journal homepage: www.elsevier.com/locate/jocs

Characterizing data patterns with core–periphery network modeling

Jianglong Yan [a,e], Leandro Anghinoni [b,*], Yu-Tao Zhu [c], Weiguang Liu [a], Gen Li [d], Qiusheng Zheng [d], Liang Zhao [c,e]

[a] *School of Computer Science, Zhongyuan University of Technology, ZhengZhou, China*
[b] *Institute of Mathematics and Computer Science (ICMC), University of Sao Paulo (USP), Sao Carlos, Brazil*
[c] *China Branch of BRICS Institute of Future Networks, ShenZhen, China*
[d] *Henan Key Laboratory on Public Opinion Intelligent Analysis, Zhongyuan University of Technology, ZhengZhou, China*
[e] *Department of Computing and Mathematics, University of Sao Paulo (USP), Ribeirao Preto, Brazil*

ARTICLE INFO

Keywords:
Data classification
Core–periphery network
Dispersed class pattern
COVID-19

ABSTRACT

Traditional classification techniques usually classify data samples according to the physical organization, such as similarity, distance, and distribution, of the data features, which lack a general and explicit mechanism to represent data classes with semantic data patterns. Therefore, the incorporation of data pattern formation in classification is still a challenge problem. Meanwhile, data classification techniques can only work well when data features present high level of similarity in the feature space within each class. Such a hypothesis is not always satisfied, since, in real-world applications, we frequently encounter the following situation: On one hand, the data samples of some classes (usually representing the normal cases) present well defined patterns; on the other hand, the data features of other classes (usually representing abnormal classes) present large variance, i.e., low similarity within each class. Such a situation makes data classification a difficult task. In this paper, we present a novel solution to deal with the above mentioned problems based on the mesostructure of a complex network, built from the original data set. Specifically, we construct a core–periphery network from the training data set in such way that the normal class is represented by the core sub-network and the abnormal class is characterized by the peripheral sub-network. The testing data sample is classified to the core class if it gets a high coreness value; otherwise, it is classified to the periphery class. The proposed method is tested on an artificial data set and then applied to classify x-ray images for COVID-19 diagnosis, which presents high classification precision. In this way, we introduce a novel method to describe data pattern of the data "without pattern" through a network approach, contributing to the general solution of classification.

1. Introduction

Over the last decades, supervised learning has advanced a lot due to the development of new techniques and the advances in hardware capacity. This learning paradigm has been applied to many real world applications that can be represented as a data classification problem. In such problem, the algorithm learns a function from the labeled samples (training data set) that maps the data to the classes, which later is used to classify the unlabeled data [1]. A number of classification techniques have been developed ([1,2]), such as the *k*NN, Naive-Bayes, MLP, SVM, Random Forest, and various Deep Learning models. However, all of them work in the following similar way: Splitting the data space or feature space into sub-spaces that best separates each class. In this scheme, strong distortions of the decision boundaries are generally not allowed. Moreover, the classifier is hard to be interpreted, specially in deep learning models, due to its automatic feature extraction nature, which hurdles the relationship between the classification results and the structure of the original data. Other than that, semantic relationships among data are not considered, which makes it unable for these classifiers to learn data patterns with complex geometrical forms and in sparse data.

Regarding the classification problem, each class of the input data sets not only present different physical features, characterized by distance, similarity or distribution among data samples, but also form different patterns with semantic meanings. The consideration of such data patterns is specially useful for classifying data items, which are not separable only by physical features. One powerful way to capture the relationship among data and, consequently, identify the data patterns, is through the topology structure of a complex network. The original

[*] Corresponding author.
E-mail addresses: 15638100054@163.com (J. Yan), anghinoni@usp.br (L. Anghinoni), zhuyutao@bifnc.cn (Y.-T. Zhu), weiguang.liu@zut.edu.cn (W. Liu), liamao1995@163.com (G. Li), zqs@zut.edu.cn (Q. Zheng), zhao@usp.br (L. Zhao).

https://doi.org/10.1016/j.jocs.2022.101912
Received 14 July 2022; Received in revised form 10 October 2022; Accepted 14 November 2022
Available online 24 November 2022

idea of building a network_based classification technique to characterize data patterns has been proposed in [3,4], where the classification component is performed by exploring the topological properties of the networks built from the training and testing data sets. Later, another network-based approach according to the “importance” concept for classification has been proposed in [5]. Although several advances have been made on semantic classification, capturing data patterns is, in general, still a hard problem.

Another salient feature, which makes the classification problem hard is the requirement of high similarity among data samples or data features within each class. However, such a hypothesis is not always satisfied, since, in real-world applications, we frequently encounter the following situation: On one hand, the data samples of some classes (usually representing the normal cases) present well defined patterns; on the other hand, the data features of other classes (usually representing abnormal classes) present large variance, i.e., low similarity within each class. One of such real world examples is the COVID-19 diagnosis by classifying X-ray chest images. The images of normal lungs present high similarity, while the images of COVID-19 present large variance among them.

In order to characterize the complex data patterns and solve the above-mentioned problem, where large dispersion may occur within each class, in this work, we explore a particular arrangement of the data, where the data features of some classes (usually representing the normal cases) present a well defined pattern and the data features of other classes (usually representing abnormal classes) present large variance, i.e., low similarity within each class. In the topology space this can be represented by a core–periphery network allowing the presence of dense and irregular sparse classes at the same time.

To exemplify the usefulness of network patterns for data classification, we consider a real-world application of X-ray chest image classification for COVID-19 diagnosis. In the X-ray image data-set, we have two classes: (i) Normal: represented by healthy lungs and (ii) COVID-19: represented by lungs infected with the virus. In this case, the samples of the normal class form a regular pattern, while the features of the samples in the COVID-19 class are so dispersed that it is hard to enclosure them in a sub-space, if not impossible. A preliminary indicator that can be used to verify that is to calculate the mean and standard deviation of the distance between the features extracted from the images. In our experiments we used four features: histogram of the image pixels, frequency components of Fourier transform, histogram of Quadtree division [6] and the fractal dimension [7] of the images. We found that the COVID-19 features were so dispersed that it would be hard to define the boundaries of this class.

The COVID-19 virus can cause acute respiratory syndrome, attacking the lungs of the person infected. Classifying the image of the lung, in a fast and precise way, is of great importance, since its an indicator of the disease severity that cannot be measured by PCR tests. One of the important COVID-19 diagnosis methods is the X-ray chest image classification [8,9]. In this paper, we will present a novel diagnosis technique, contributing to this type of evaluation.

This paper is an extended version of the paper previously published in conference proceedings [10]. Here, we have largely enhanced the theoretical and technological analyses and discussions of the method through presenting a number of new materials, making its innovation much clearer. We present a novel classification technique, where the normal class is represented by the core and the abnormal class is represented by the periphery. In the classification phase, we calculate the coreness measure of each testing sample. A high coreness value classifies the testing sample to the normal class; otherwise, it belongs to the abnormal class.

2. Related work

Our technique is based on network structures, that can represent many real systems. Although the state-of-the art methods are very effective in image classification, the relationship between the classification results and the original data is hard to be interpreted and to be understood. Our method, on the other hand, is more transparent, as we can visualize the relationship in the graph and calculate network measures to understand the original data. Each network measure has a transparent meaning. If the network has a high value of the coreness measure, it means that the underlying data contains a structure with a well-connected normal class and an ill-posed and dispersed abnormal class. Other network measures can be considered to be used in our approach too. For example, a high value of the assortativity measure implies in high homogeneity of the data; a high value of the clustering coefficient implies in local sub-groups; a high modularity implies in well-defined communities, and so on. In summary, the final classification relies on the network structure constructed from the original data and characterized by networks measures. Each network measure has a transparent meaning. Therefore, the classification results are, in large extent, transparent.

Complex network refers to large scale graphs with nontrivial connection patterns [11,12]. Its mesoscale structure, such as community and core–periphery structure, can be very useful to discover relationships in the data. Over the last decades a lot of attention has been given to the community structure of data, yielding many powerful algorithms for community detection [13]. However, the community structure also implies a high similarity among data samples in the same community.

The core–periphery structure, on the other hand, implies a high similarity only among core nodes, which is more suitable to the problem we aim to solve. A core–periphery network consists of a subset or subsets of strongly connected nodes to form cores and a subset or subsets of low degree peripheral nodes. Usually, the peripheral nodes are connected to the cores, but have few or even no connection to other peripheral ones. Several works have been dedicated to this type of network and this structure can be found in many complex systems [14–18].

Since the seminal work by Borgatti and Everett [14], many other methods have been proposed to detect core–periphery structures in networks [19–24]. The first method proposed by Borgatti and Everett [14] is based on an idealized pattern matrix that can be optimized to sort the nodes into core and periphery. The following works treated several caveats raised in the original method, such as the lack of statistical significance of the core–periphery structure found by the method and introducing quality measures to optimize the size of each block (core and periphery). Holme [17] tackles the statistical significance issue by introducing null models into the proposed algorithm, that is based on the closeness centrality measures of the core and periphery. Later, Kojaku and Masuda [25] argues that a core–periphery structure should not be judged merely by the degree of the node, which the previous works tend to do, and proposes an algorithm that takes into account three blocks instead of two. A very recent work [24], however, has proposed a fast and exact greedy algorithm to solve the single core detection task. The author presents a numerical proof of their solution and carries out real applications.

The research naturally evolved to find multiple core–periphery structure. In [22], the authors present a scalable algorithm for this task, which had been treated before but was very costly. More recently, the research in this area has led to works that investigates what really configures a core–periphery structure and how this mesostructure can be compared to a community structure.

These works provide a strong background on how to mathematically deal with these types of structure, however they ultimately provide unsupervised clustering methods for network systems. As far as we know, using the core–periphery structure as a network_based classification framework presents a novelty in supervised learning.

Another technique used to represent data in the form of graphs are knowledge graphs [26]. This type of graphs is applied to model the relationship between nodes or groups of nodes within the graph, where each node is an entity description (an object, a word, an idea,

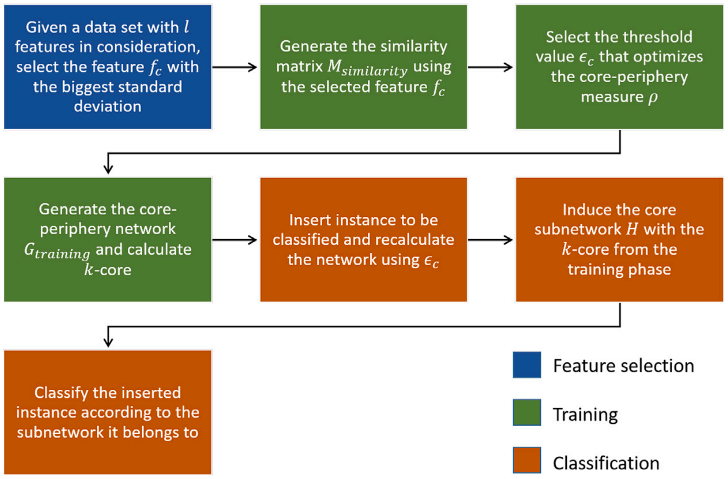

Given a data set with l features in consideration, select the feature f_c with the biggest standard deviation
Generate the similarity matrix $M_{similarity}$ using the selected feature f_c
Select the threshold value ϵ_c that optimizes the core-periphery measure ρ
Generate the core-periphery network $G_{training}$ and calculate k-core
Insert instance to be classified and recalculate the network using ϵ_c
Induce the core subnetwork H with the k-core from the training phase
Classify the inserted instance according to the subnetwork it belongs to
Feature selection
Training
Classification

Fig. 1. Feature selection (blue): in this step we select the feature that will be used for classification, based on the dispersion of the features (the feature with the highest variance is selected). Training (green): in these three steps we construct a network that optimizes the core–periphery measure ρ. Classification (orange): in these final steps we insert the instance to be classified in the trained network and attribute a class to this instance.

etc.). The nodes are interlinked based on the ontology of the data. The relationship among nodes, such as hierarchy, class, causality, type, etc. can be inferred from the graph. In general, knowledge graphs consider the relationship among nodes within graph and, each time, part of the graph (a subset of nodes and their corresponding links) are involved in a specific inference. On the other hand, our approach considers the whole graph structures and we look for the original data patterns by means of characterizing the whole graphs constructed from the original data. In this work, we characterize in which degree the test data samples (nodes) satisfy the constructed core–periphery network.

3. Methods

In this section we detail the proposed method, which uses a core–periphery structure to represent the two classes of the data. The core represents the data with high similarity and low dispersion, while the periphery represents the class with dispersed features. In the case of COVID-19, that will be explored in one of the applications, the core is the normal class (healthy lungs) and periphery is the COVID-19 class (lungs infected by COVID-19). Although we present here a method for binary classification, the idea can be extended to multi-class classification, as will be discussed in the Conclusions section.

Fig. 1 shows the overall process and the main steps of the proposed method. In the first step (blue), we select the feature for classification, based on the standard deviation of the features (the feature with the highest variance is selected. That is done because the most disperse feature helps to construct the core–periphery network, which is the main point of this work. This is analogous to a PCA analysis, where we want to preserve the feature that explains the most variance in the data. Yes, we can select more than one features with the highest standard deviations. However, our numerical study shows that the selection of the one with the highest standard deviation always leads to good classification results besides of the lower computation loads. In the three next steps (green) we construct a network that optimizes the core–periphery measure ρ. In the final steps (orange) we insert the instance to be classified in the trained network and attribute a class to this instance.

3.1. Feature extraction phase

This step of the method can vary depending on the type of data that is being treated. The general model considers a set of n samples used for training $X_{training} = \{(x_1, y_1), \dots, (x_n, y_n)\}$, where the first component of the ith tuple $x_i = \{f_1, \dots, f_d\}$ denotes the d-dimensional features of the ith training instance (note that each feature f_i can have different dimensions and x_i is the concatenation of all these features). The second component of the tuple y_i denotes the class label of the sample. Here we treat the classification as binary problem, therefore, $y_i = 1$, if the sample x_i belongs to core sub-network and $y_i = 0$, if it belongs to the periphery sub-network.

For the training phase, only one of the d-dimensional features is selected. To do so, for each feature f, the standard deviation σ_i is calculated. Finally, the feature that present the highest standard deviation f_c is selected to represent the data samples in the training and classification process, i.e.,

$$f_c = argmax(\{\sigma_1, \dots, \sigma_l\}) \tag{1}$$

where l is the number of features under consideration. For example, in this paper, $l = 4$ in the real application, because the following 4 features are considered: pixel histogram, components of Fourier transform, fractal dimension and histogram of Quadtree division.

We will see in the artificial example that the features are generated artificially in order to validate the model. In this case the feature selection step is not performed. In the COVID-19 application the dataset is composed of images and features extraction is performed as following.

For each medical image, we extract the following features: (1) Pixel histogram; (2) frequency components of Fourier transform; (3) fractal dimension [7,27]; (4) histogram of Quadtree division [6]. The two former features reveals the statistical properties of the images, while the later ones characterize geometrical complexity of the images.

As we will see in the next section, the healthy lungs present high similarity of the four features, while the similarity among the lungs with COVID-19 is very low. In order to characterize the dispersed pattern of COVID-19 images, One of the features with the biggest average variance is chosen to construct the core–periphery network.

3.2. Training phase

In the training phase, a core–periphery network $G_{training}$ is constructed using the selected image features of the training set. After calculating a feature vector for every data sample and using the feature with the largest standard deviation f_c, a similarity matrix $M_{similarity}$ is formed containing the Euclidean distance between each pair of data features. We have compared the average distance and standard deviation using different distance measures – cosine and canberra – and observed that the main characteristics are preserved, i.e. high dispersion for the covid class and lower for the normal class. Therefore, we concluded that the distance measure selection is not critical to the network construction and, consequently to the classification result. Each image is a node in $G_{training}$ and the connection between a pair of nodes is made if the distance in $M_{similarity}$ is smaller than a specific value of ϵ. We optimize ϵ (see Algorithm 1) based on the normalized ρ measure, that evaluates the core–periphery structure according to Eq. (2) [14]. When this measure reaches its maximum we select the final network $G_{training}$ and threshold ϵ_c. In all experiments, we set $step = 0.01$.

Algorithm 1 ρ optimization.

```
Require: $c = [c_1, c_2, ..., c_n]$                      ▷ Coreness vector
Require: $M_{similarity}$                                ▷ Similarity matrix
  $\epsilon \leftarrow 0$
  $i \leftarrow 0$
  while $\rho_i > \rho_{i-1}$ do
      $A \leftarrow M_{similarity}$, where $M_{similarity} < \epsilon$;
      $\rho = \frac{1}{n^2} \sum_{i,j} a_{ij}\delta_{ij}$;
      $\epsilon \leftarrow \epsilon + step$
      $i \leftarrow i + 1$
  end while
  return $\epsilon_c$
```

$$\rho = \frac{1}{n^2} \sum_{i,j} a_{ij}\delta_{ij}, \tag{2}$$

where A is the adjacency matrix of network $G = (V, E)$ with n nodes and its element $a_{ij} = 1$ if node i and node j are linked and 0, otherwise. $\delta_{ij} = c_i c_j$, where c_i measures the coreness of the node i, $c_i = 1$ or $c_i = 0$ means that node i belongs to the core or the periphery, respectively. In other words, the coreness vector is the target label vector. Specifically, ρ measure reaches the maximum value when $A = \Delta = [\delta_{ij}]$, i.e., the core–periphery structure seeks to find out a membership vector $\mathbf{c}$ to maximize ρ. The first term $(\frac{1}{n^2})$ is a normalization factor, since n^2 is the maximum possible value for ρ.

Due to the sparsity of the periphery nodes, the optimum threshold value ϵ_c can generate a network with unconnected nodes (singletons). To overcome this situation we use kNN to force the node to connect to its closest neighbors. Therefore, the neighborhood of a given vertex f_{c_i} expressed in the adjacency matrix $M_{training}$ is given by:

$$M_{training_{ij}} = \begin{cases} 1, & \text{if } d(f_{c_i}, f_{c_j}) < \epsilon_c \\ kNN(f_{c_i}), & \text{if } f_{c_i} \text{ is a singleton} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

where $kNN(f_{c_i})$ returns the closest vertex to f_{c_i} (we use $k = 1$) and $M_{training}$ is the final adjacency matrix to be used in the following steps. Obtaining the adjacency matrix $M_{training}$, we automatically get the training core–periphery network $G_{training}$.

3.3. Classification phase

At the classification phase, we insert the testing data sample x_{test}, using the selected feature $f_{c_{test}}$, to the core–periphery network constructed so far ($M_{training}$) and generate a new matrix M_{test}, where each element is the distance between each pair of training and the test data features. In other words, M_{test} is the same as $M_{training}$, but it has one more line and column than $M_{training}$ regarding the test data feature $f_{c_{test}}$. We use the same ϵ_c and feature f_c selected in the previous step. Then we construct G_{test} from the adjacency matrix M_{test}. Finally, we induce a sub-network H, which is the k-core of G_{test} with $k = K_c$-core (K_c-core is the value of the k-core of $G_{training}$). The k-core [18,28] of a network is the maximal component in which all vertices have a degree of at least k. In the core–periphery structure, there is a K_c-core containing the nodes of the normal class (core). To classify a new instance we verify if it belongs to H (core) or not (periphery).

$$y_{test} = \begin{cases} 1, & \text{if } x_{test} \text{ belongs to } H \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

where y_{test} is the class of x_{test} and H is the k-core of G_{test} with $k = K_c$-core.

Specifically, we perform the following steps for classification (also detailed in Algorithm 2):

1. Add the new instance representation, using the selected feature $f_{c_{test}}$ to the data-set;
2. Calculate the distance matrix $M_{similarity}$, considering all the training data features and the test data feature;
3. Use the ϵ_c value obtained in the training phase to generate the adjacency matrix M_{test} and its corresponding network G_{test};
4. Extract a sub-graph H with $k - core$ equals to the one of the original network (K_c-core);
5. If the new instance belongs to the sub-graph H it is classified as core, otherwise classify it as periphery.

Algorithm 2 New instance classification

```
Require: $f_{c_{test}}$                          ▷ New instance feature vector
Require: $\epsilon_c$                            ▷ Optimum threshold value
  $H \leftarrow \emptyset$
  Calculate new $M_{similarity}$ ▷ Considers training data and new instance
  $M_{test} \leftarrow M_{similarity}$, where $M_{similarity} < \epsilon_c$
  Construct $G_{test}$ from $M_{test}$
  $H \leftarrow K_c$-core of $G_{test}$
  if $x_{test} \in H$ then:
      $y_{test} \leftarrow 1$                    ▷ Core class
  else:
      $y_{test} \leftarrow 0$                    ▷ Periphery class
  end if
  return $y_{test}$
```

4. Numerical analysis and computer simulation results

In this section, we present numerical results to show the salient features of the proposed classification method.

4.1. Simulations on artificial data set

Here we present a simple toy model to replicate the steps of the proposed method and visualize the expected results and the key advantages of the proposed method. The data-set is composed of 220 2-dimensional vectors (Fig. 2). The core (red dots) is composed of 60 samples, while the periphery (blue crosses) is composed of 160 samples. The core forms a dense circle, with the samples evenly distributed around the center of the image at coordinates $(0.5, 0.5)$. The periphery is represented by four concentric circles, that are not fully connected (the lower part of the circles are missing). The green triangle is the test data sample to be classified.

We can also observe that the toy data presents the characteristics mentioned before for a core–periphery model, i.e., the mean distance and standard deviation is low for the core samples and high for periphery samples (Table 1). Another point to note is that we do not perform feature selection in this toy model, since the data was generated for visualization purposes and presents a core–periphery structure.

We will see in the experiments that the missing data in the lower part poses a challenge to traditional classification methods. This was intended in the selection of the toy model for two main reasons: (i)

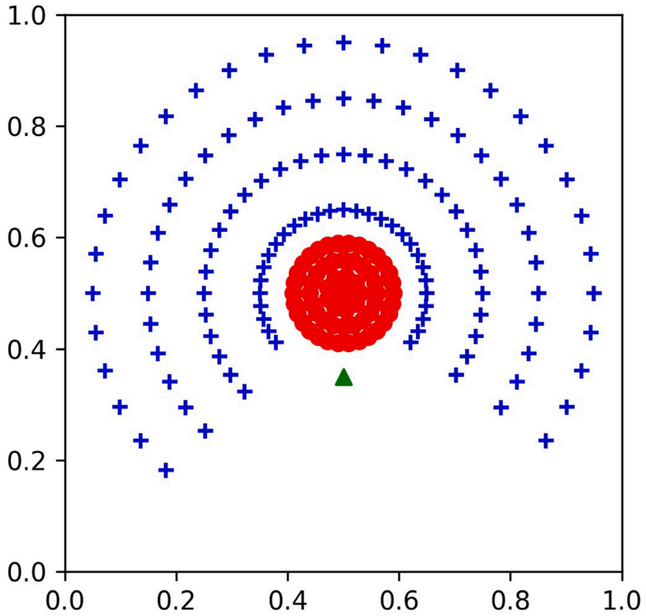


Fig. 2. 2-dimensional artificial data used to test the model.

Table 1
Average distance and standard deviation of the training artificial samples.

	Mean distance		Standard deviation	
	Core	Periphery	Core	Periphery
Artificial samples	0.104	0.432	0.049	0.221

in the core–periphery model, the core should be correctly detected no matter the pattern of the periphery and (ii) peripheral data (dispersed) can be often incomplete, specially in high dimensional features. In the COVID-19 experiment, for example, we may not have seen all types of variations, what can lead to these empty areas in the n-dimensional space.

4.1.1. Core–periphery network formation — training

In this step, we calculate a similarity matrix $M_{similarity}$ using all the 2 dimensional features of the 220 samples. This matrix is shown in Fig. 3

Next, we use Algorithm 1 to find the value for ϵ that maximizes the ρ measure. This is done by increasing the value of ϵ until the aggregation of a new node to the core has no impact in the value of ρ. The optimization process is depicted in Fig. 4. For this case, the highest value for ρ is 0.1147 and the threshold value of ϵ_c is 0.201.

Finally, we can generate the core–periphery adjacency matrix $M_{training}$ and the network $G_{training}$ that will be used for classification. In this step, all the entries in the distance matrix that are above the ϵ_c value are set to zero. We also do not consider self-loops in the network. The constructed network $G_{training}$ presents the following basic info: (i) Number of nodes: 187; (ii) number of edges: 4547; (iii) average degree: 48.62 (overall); (iv) average degree: 84,58 (core); (v) average degree: 30.21 (periphery). These two structures are shown in Fig. 5

4.1.2. Classification results

In this step, we are going to classify the test sample (green triangle) depicted in Fig. 2. The green triangle data point is a continuation of the smallest peripheral circle. A network-based classification should take this feature into account and classify this instance as a periphery sample, due to its ability of classification by pattern formation.

For that, first we calculate the $k-core$ of the sub-graph H induced by the core samples in the trained network. The value obtained for this data-set was 62, meaning that the core is fully connected (each node is connected to the other 62 nodes belonging to the 63 node core). Then the new instance is classified using the method presented in Section 3.

Fig. 6 shows, respectively, the classification of the green triangle (which was classified as periphery) in the topological domain and in the Cartesian plane.

In order to compare the classification performance of our method, we ran the classification of the test sample using seven different classifiers (AdaBoost (Ada), Decision Tree (DT), Multi-Layer Perceptron (MLP), kNN (KNN), Random Forest (RF) and Support Vector Machine (SVM)). Since these traditional methods rely on the physical features of the data, the missing part of the concentric circles imposes a challenge for them and none can correctly classify the green triangle as a peripheral sample. The decision boundaries of each method is depicted in Fig. 7.

As we can see, although a high level evaluation of this image may lead to the conclusion that part of the concentric blue circles are missing and that the triangle is part of one of them, the traditional classifiers, that split the space into sub-spaces cannot take that into consideration and end up by extending the core to this empty area.

From the simulation results presented above, we perceive the following features of the proposed model:

1. The proposed classification method considers not only the physical features or class typologies of the data, but also considers the semantic organization of the data, therefore, it is able to classify data sample according to semantic pattern formation of the data;
2. It can identify the data class (periphery class) even without a well defined physical pattern; on the other hand, other classic classification techniques may fail in such kind of situations.
3. The peripheral samples can have distinct distributions.

4.1.3. Comparison of the proposed method with outlier detection methods

At first glance, the proposed method may seem to share characteristics with outlier detection methods. However, the proposed method has basic differences to outlier detection [29–31]. Outliers are usually

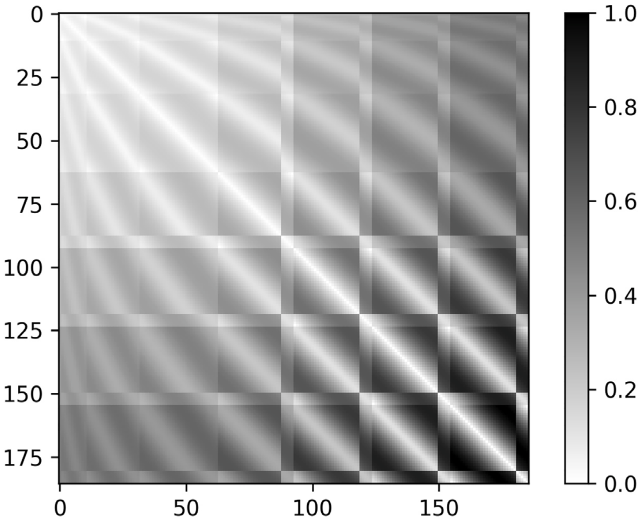


Fig. 3. Similarity matrix for the artificial data-set.

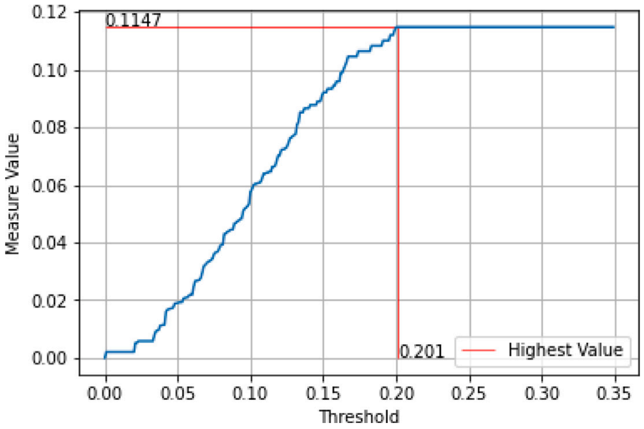


Fig. 4. Optimization of ρ for the artificial samples.

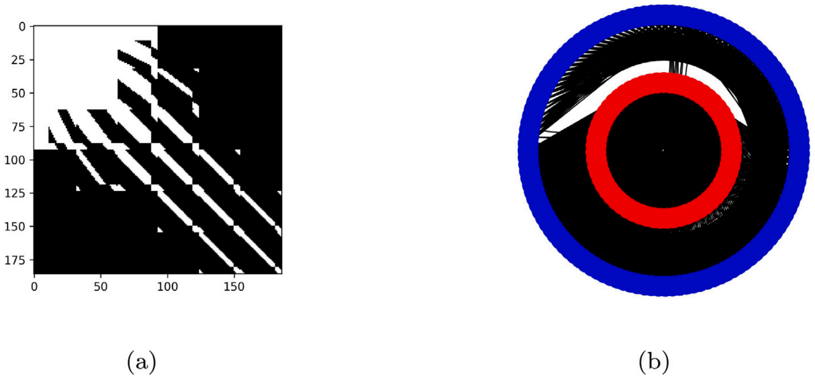


Fig. 5. Adjacency matrix and its respective core–periphery network. (a) Binary adjacency matrix with $\epsilon_c = 0.179$ (white represents a connection, i.e., 1). (b) Core–periphery network, with the core (normal class) colored in red.

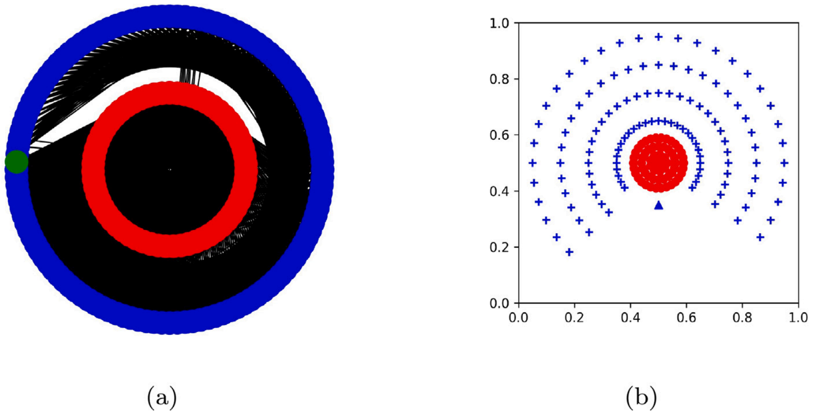


Fig. 6. (a) Network with the periphery sample inserted (green triangle). (b) Data-set with the periphery sample classified (now a blue triangle).

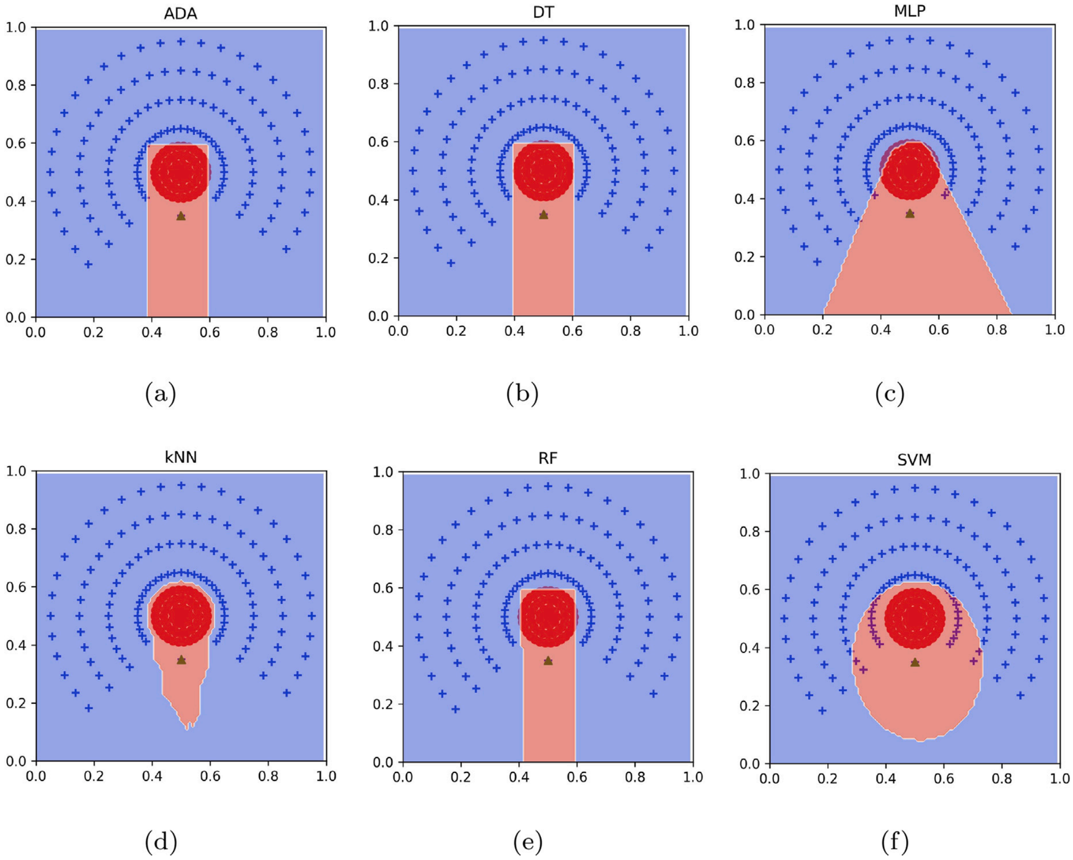


Fig. 7. Decision boundaries of traditional classifiers. (a) Adaboost. (b) Decision Tree. (c) Multi-layer Perceptron. (d) kNN. (e) Random Forest. (f) Support Vector Machine. Notice the position of the green triangle.

rare events, while the abnormal class in the core–periphery model can contain a large number of data items. Also, in the outlier detection problem, we aim to find only the pattern of the normal class, not giving any relevance to the outliers, since they are not considered a class by definition.

In this sense, the method proposed in this paper not only provides a novel classification strategy, but also can be used for outlier detection and even outlier data pattern characterization. On the other hand, outlier detection methods cannot be directly applied for classification in general case, i.e., separate the periphery from the core in this paper.

To visualize this point we have applied three different outlier detection methods to the artificial data-set. This data-set has more samples in the periphery than in the core. This is already a problem for outlier

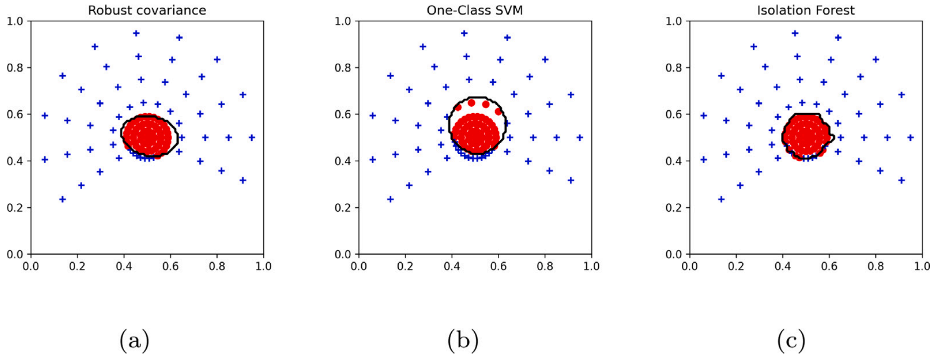


Fig. 8. Decision boundaries of outlier detection methods. (a) Robust Covariance (7 instances misclassified). (b) One-Class SVM (16 instances misclassified). (c) Isolation Forest (7 instances misclassified).

detection techniques, that assume that the outlier samples cannot outnumber the inliers. Therefore, the number of points in the periphery is reduced in this example, so that the ratio of outliers stays in 0.50 (the upper limit of some of the methods). Fig. 8 shows the results:

Notice that none of these methods could separate the two classes correctly, even when relaxing the definition of an outlier (considering half of the data-set points are outliers). On the other hand, the proposed technique can correctly detect all the "outliers" as periphery nodes, as shown by the simulation results in the last subsection.

4.2. Simulations on COVID-19 data set

Here, we present the numerical results applying the proposed technique for real X-ray chest image classification.

For this purpose, the following public data set is used: the COVID-19 Chest X-ray Database of COVID-19 Radiography Database [32]. The Kaggle Lung Images data-set contains 3.616 images labeled as 'Covid', 6.012 images labeled as 'Lung Opacity', 10.192 images labeled as 'Normal' and 1.345 images labeled as 'Viral Pneumonia'. In our work we used batches of 150 images sampled from the 'Covid' and 'Normal' data-set, so that the experiments were always balanced.

To understand the complexity of the problem, we have performed a supervised UMAP visualization of the raw images with the following parameters: neighbors = 3, components = 2. We sampled 1000 images from each class (Normal-red and Covid-blue). In Fig. 9 we can see that the raw data is highly mixed and both classes are dispersed and form irregular clusters. This emphasizes the need to select a feature that explains the most variance in data.

4.2.1. Characterizing pattern dispersion of COVID-19 images

Here, we extract features of the selected image data set. We find that the features of the normal lung images present high similarity, while the features of the images infected by COVID-19 disperse a lot. Therefore, the traditional classification hypothesis, which requires high similarity within each class, is violated in this case.

In Fig. 10, we show, respectively, four images of healthy lungs and four images of lungs infected by COVID-19. Figs. 11 and 12 show the Quadtree division and their corresponding quadrant size histogram for each of the 2 × 4 images. We see that the images of normal lungs present similar features, but the COVID-19 images possess large difference among them. Fig. 13 shows the fractal dimensions calculated for the four normal lung and the four COVID-19 images. Again, we see that the COVID-19 fractal dimension curves are much dispersed. The histograms the frequency components of Fourier transform presents the same feature.

In order to evaluate the dispersion of the features extracted, we measure the mean distance and standard deviation of each class (normal and COVID-19) in the training set (Table 2). These numbers indicate that using similarity measures to classify the COVID-19 test instances may not be possible, since this class is very dispersed (high standard deviation). In the traditional models, each class should be contained in a region of the subspace under consideration, therefore, in this case, a network-based classifier (core–periphery network model) may come as a promising solution.

Table 2
Average distance and standard deviation of the training images.

	Mean distance		Standard deviation	
	Normal	COVID-19	Normal	COVID-19
Original images	0.135	0.656	0.150	0.673
FFT	0.088	0.611	0.139	0.936
Quadtree	0.574	0.834	0.309	0.502
Fractal dimension	0.196	0.45	0.105	0.273

4.2.2. Core–periphery network formation — training

The training step consists of constructing a series of core–periphery networks with the training data. This is done by considering each image feature as a node in the network and connecting every node with distance smaller than ϵ. The goal in this step is to find the ϵ value that maximizes the function Eq. (2). Since the function is strictly increasing, we increase ϵ until the ρ-measure reaches its maximum. The trained core–periphery network is the one constructed with this final ϵ.

In Fig. 14 we can see the optimization of the ρ-measure as a function of ϵ. The ρ-measure peaks at $\epsilon = 1.14$. The trained network, constructed with this ϵ_c value is shown in Fig. 15, which presents a clear core–periphery pattern.

4.2.3. Classification results

After the core–periphery network is constructed, we make classification of each test sample according the classification algorithm. The testing data item will belong to the core class if its coreness value is high; otherwise, it is classified to the periphery class (COVID-19 class).

Our model, denoted as (CP), was tested against seven classification techniques: AdaBoost (Ada), Decision Tree (DT), Multi-Layer Perceptron (MLP), Naive-Bayes (NB), Random Forest (RF), Support Vector Machine (SVM) and Deep Convolutional Network (DCN). We obtained the classification results in two different configurations of the training and testing data sets, which are shown in Tables 3 and 4. As we can see, the proposed technique presents high classification accuracy and f1-score and low standard deviation within the runs, in comparison to the classic and the state-of-the-art techniques.

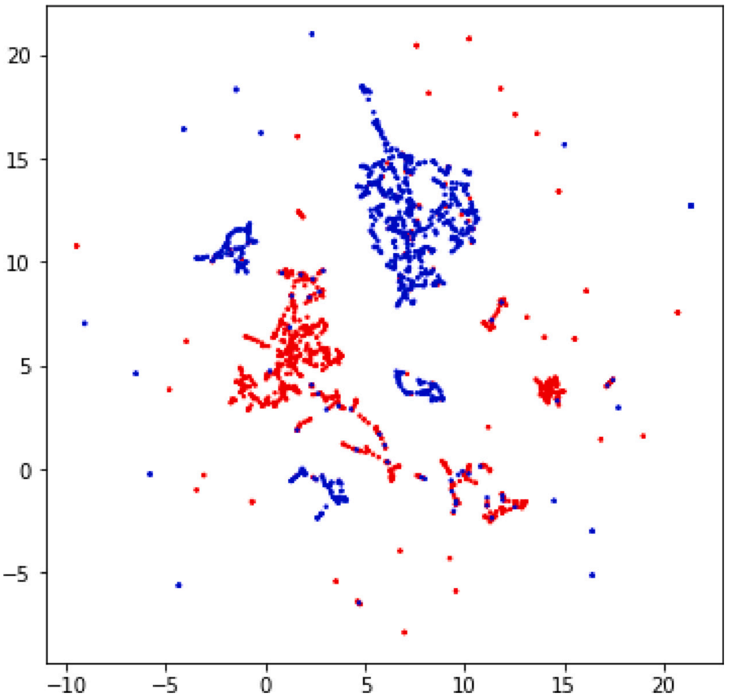
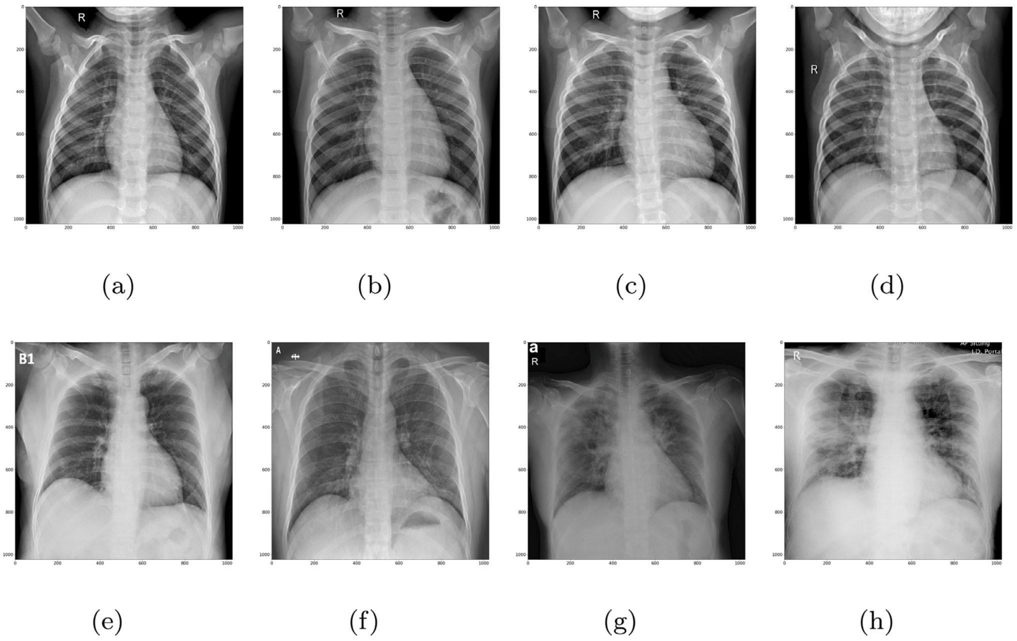


Fig. 9. Supervised UMAP embedding of 1000 Normal samples (red) and 1000 Covid samples (blue).

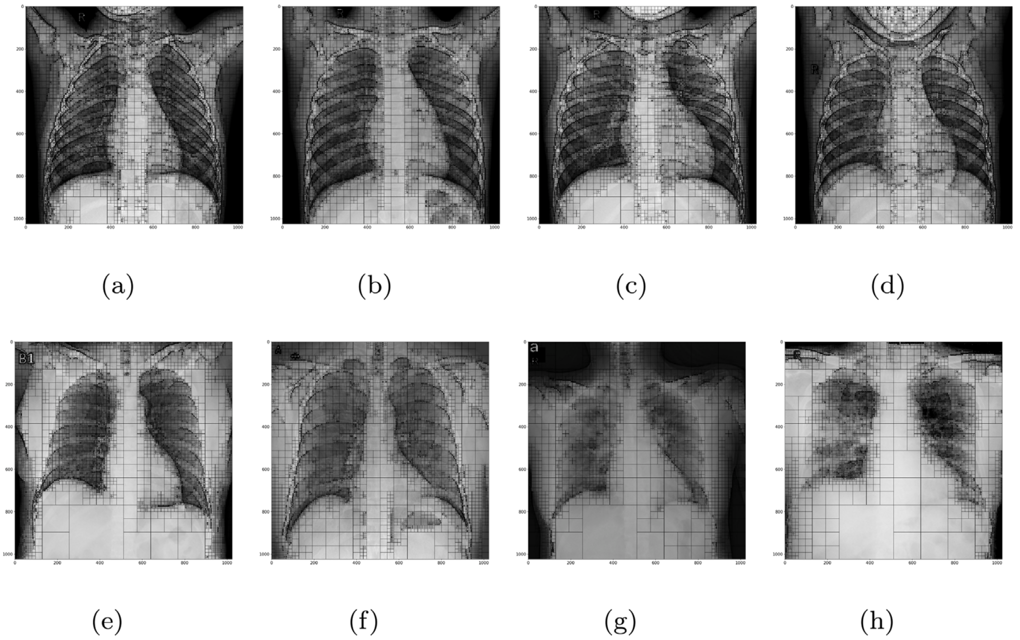

(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 10. (a)–(d): Four healthy lung images. (e)–(h): Four images infected by COVID-19.
Source: Data source: COVID-19 Radiography Database [32] (https://www.kaggle.com/tawsifurrahman/covid19-radiography-database).

5. Conclusions

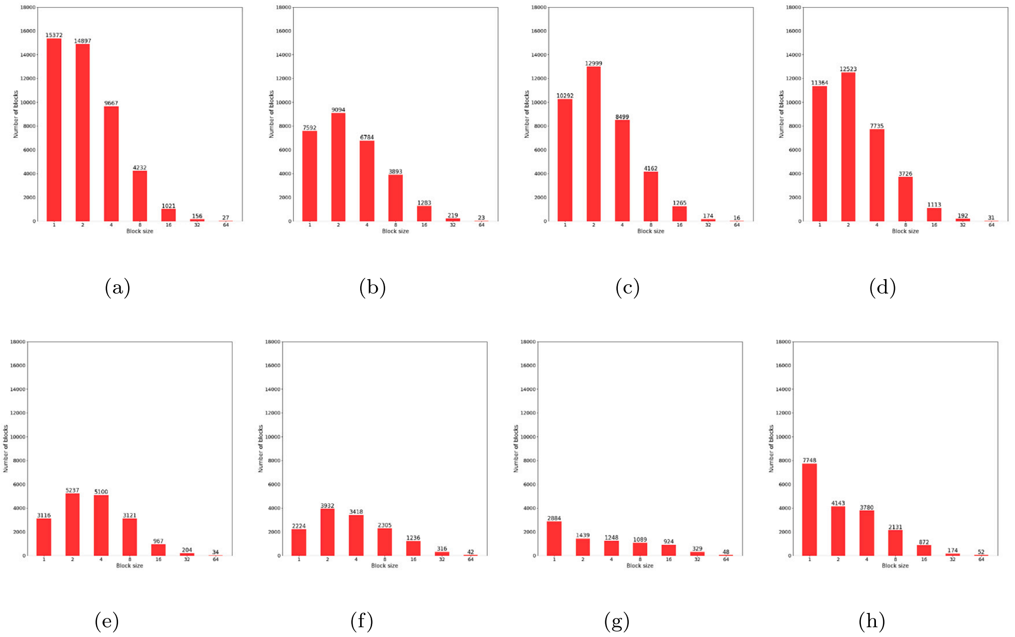
In this work, we have presented a novel method to handle data classification problems when the training set is not separable in the physical space, as in the classic classification paradigm. We argued that in certain situations the context and semantic of the data has to be taken into consideration in order train the classifier properly. To do so, we proposed converting the data into a core–periphery network structure, where the core represents the normal data, or the data with a pattern, and the periphery represents the dispersed data, or the data



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 11. Each figure is the Quadtree division of the corresponding image shown by Fig. 10.



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 12. Each figure is the histogram of the quadrant sizes of the corresponding Quadtree division shown by Fig. 11.

without a pattern. In spite of its simplicity, this approach can shed some light on problems where the physical separability is an issue.

We have shown in the experiments that the method performs well both in a toy example and in a real problem. In the toy model we showed that the traditional methods may not be able to evaluate high level patterns and tend approximate empty areas in the physical space to the closest class, regardless of the data pattern formation. This was expected, since these methods are based on the division of the space into sub-spaces and, therefore, they cannot imply the continuation of a high level pattern if there are no samples in the training set.

As a future work, we believe a similar approach can be used for multi-class problems, by the use of multi-core networks. A common

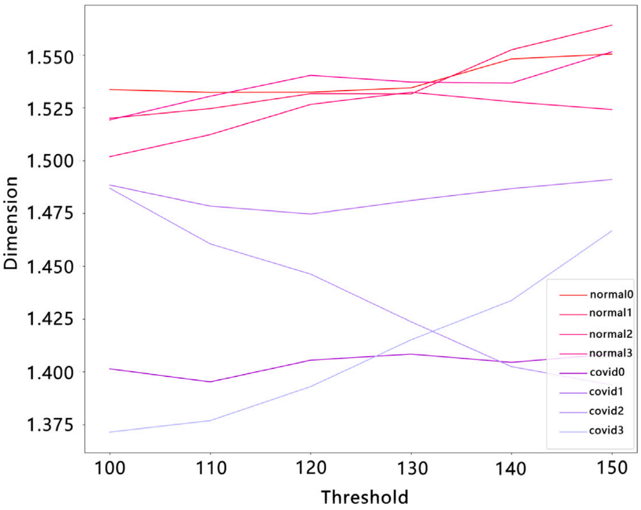


Fig. 13. Fractal dimensions against the binary image thresholds of the images shown by Fig. 10. The box-counting dimension is calculated on each binary image generated from the original image with a specific threshold value.

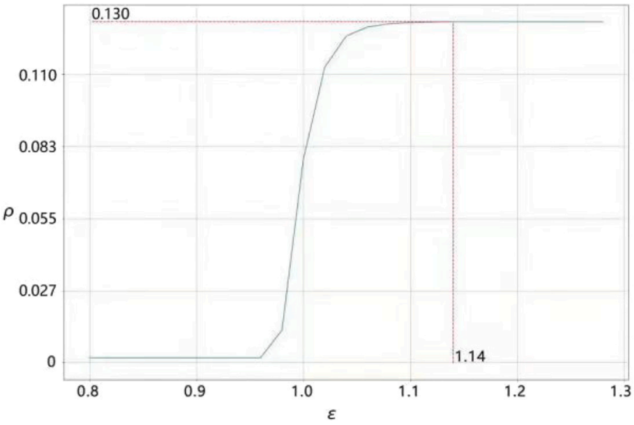
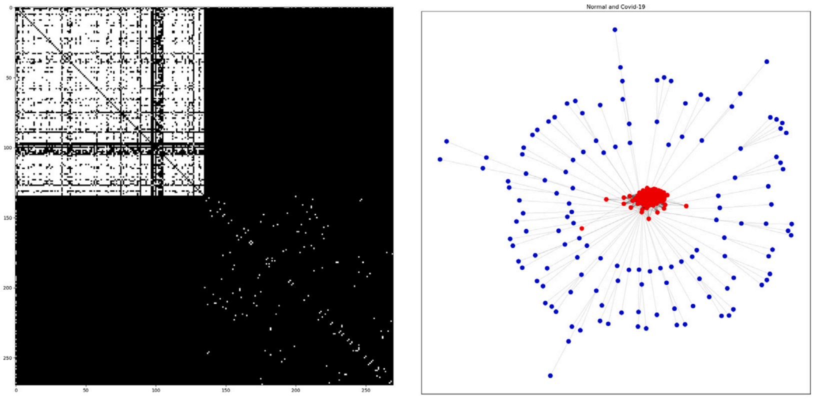


Fig. 14. ρ-measures of the networks generated by varying the ϵ values.



(a) (b)

Fig. 15. Adjacency matrix and its respective core–periphery network. (a) Binary adjacency matrix with ϵ_c = 1.14 (white represents a connection, i.e., 1). (b) Core–periphery network, with the core (normal class) colored in red.

Table 3
Classification comparison with traditional methods. This table shows the results using a 10% split of the data-set (270 images for training and 30 images for testing). The experiment was run 50 times. For each run, we randomly generate a training set and a testing set with fixed size (270 and 30, respectively). We report the average and the standard deviation of the accuracy and f1-score. The best results are highlighted in bold.

Technique	Acc.Mean	Acc.StDev	f1.Mean	f1.StDev
DCN	0.82	0.06	0.79	0.06
SVM	0.86	0.07	0.88	0.06
DT	0.90	0.07	0.90	0.07
RF	0.91	0.05	0.91	0.05
MLP	0.91	0.08	0.92	0.07
ADA	0.93	0.05	0.93	0.06
NB	0.93	0.04	0.93	0.04
CP	**0.97**	**0.02**	**0.97**	**0.01**

Table 4
Classification comparison with traditional methods. This table shows the results using a 20% split of the data-set (240 images for training and 60 images for testing). The experiment was run 50 times. Again, for each run, we randomly generate a training and a testing set maintaining their respective sizes. We report the average and the standard deviation of the accuracy and f1-score. The best results are highlighted in bold.

Technique	Acc.Mean	Acc.StDev	f1.Mean	f1.StDev
DCN	0.81	0.04	0.81	0.04
SVM	0.85	0.05	0.87	0.04
DT	0.87	0.06	0.87	0.06
RF	0.90	0.05	0.90	0.04
MLP	0.89	0.04	0.89	0.04
ADA	0.91	0.04	0.91	0.04
NB	0.92	0.04	0.92	0.04
CP	**0.92**	**0.01**	**0.92**	**0.01**

way to solve such problems in the topological space is to use community detection and cluster each class in a community. This approach, however, may fail to capture the dispersed samples, that are usually clustered to the closest community. A multi-core structure would, in theory, be able to consider different levels of dispersion and aggregate this samples following their pattern formation.

CRediT authorship contribution statement

Jianglong Yan: Conceptualization, Methodology, Software, Writing – original draft, Writing – review & editing, Formal analysis, Validation. **Leandro Anghinoni:** Conceptualization, Methodology, Software, Writing – original draft, Writing – review & editing, Formal analysis, Validation. **Yu-Tao Zhu:** Conceptualization, Methodology, Writing – original draft. **Weiguang Liu:** Conceptualization, Methodology, Writing – original draft. **Gen Li:** Software, Writing – review & editing. **Qiusheng Zheng:** Conceptualization, Methodology, Writing – original draft. **Liang Zhao:** Conceptualization, Methodology, Writing – original draft, Writing – review & editing, Formal analysis, Validation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The link to the data used in our research is available in the manuscript.

Acknowledgments

This work was carried out at the Center for Artificial Intelligence (C4AI-USP), with support by the São Paulo Research Foundation (FAPESP) under grant number: 2019/07665-4 and by the IBM Corporation. This study was also financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) – Finance Code PROEX-3314750/D.

References

[1] C.C. Aggarwal, C.R.D. Clustering, Algorithms and applications, 2014.

[2] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444.

[3] T.C. Silva, L. Zhao, Network-based high level data classification, IEEE Trans. Neural Netw. Learn. Syst. 23 (6) (2012) 954–970.

[4] T.C. Silva, L. Zhao, High-level pattern-based classification via tourist walks in networks, Inform. Sci. 294 (2015) 109–126.

[5] M.G. Carneiro, L. Zhao, Organizational data classification based on the importance concept of complex networks, IEEE Trans. Neural Netw. Learn. Syst. 29 (8) (2017) 3361–3373.

[6] R.A. Finkel, J.L. Bentley, Quad trees a data structure for retrieval on composite keys, Acta Inform. 4 (1) (1974) 1–9.

[7] K. Falconer, Fractal Geometry: Mathematical Foundations and Applications, John Wiley & Sons, 2004.

[8] Y. Hu, J. Jacob, G.J. Parker, D.J. Hawkes, J.R. Hurst, D. Stoyanov, The challenges of deploying artificial intelligence models in a rapidly evolving pandemic, Nat. Mach. Intell. 2 (6) (2020) 298–300.

[9] M.A. Elaziz, K.M. Hosny, A. Salah, M.M. Darwish, S. Lu, A.T. Sahlol, New machine learning method for image-based diagnosis of COVID-19, PLoS One 15 (6) (2020) e0235187.

[10] J. Yan, W. Liu, Y.-t. Zhu, G. Li, Q. Zheng, L. Zhao, Classification of dispersed patterns of radiographic images with COVID-19 by core-periphery network modeling, in: International Conference on Complex Networks and their Applications, Springer, 2021, pp. 39–49.

[11] A.-L. Barabási, R. Albert, Emergence of scaling in random networks, Science 286 (5439) (1999) 509–512.

[12] D.J. Watts, S.H. Strogatz, Collective dynamics of 'small-world' networks, Nature 393 (6684) (1998) 440–442.

[13] X. Su, S. Xue, F. Liu, J. Wu, J. Yang, C. Zhou, W. Hu, C. Paris, S. Nepal, D. Jin, et al., A comprehensive survey on community detection with deep learning, IEEE Trans. Neural Netw. Learn. Syst. (2022).

[14] S.P. Borgatti, M.G. Everett, Models of core/periphery structures, Social Networks 21 (4) (2000) 375–395.

[15] P. Csermely, A. London, L.-Y. Wu, B. Uzzi, Structure and dynamics of core/periphery networks, J. Complex Netw. 1 (2) (2013) 93–123.

[16] S.N. Dorogovtsev, A.V. Goltsev, J.F.F. Mendes, K-core organization of complex networks, Phys. Rev. Lett. 96 (4) (2006) 040601.

[17] P. Holme, Core-periphery organization of complex networks, Phys. Rev. E 72 (4) (2005) 046111.

[18] S. Carmi, S. Havlin, S. Kirkpatrick, Y. Shavitt, E. Shir, A model of internet topology using k-shell decomposition, Proc. Natl. Acad. Sci. 104 (27) (2007) 11150–11154.

[19] F.D. Rossa, F. Dercole, C. Piccardi, Profiling core-periphery network structure by random walkers, Sci. Rep. 3 (1) (2013) 1–8.

[20] A.R. Benson, D.F. Gleich, J. Leskovec, Higher-order organization of complex networks, Science 353 (6295) (2016) 163–166.

[21] P. Rombach, M.A. Porter, J.H. Fowler, P.J. Mucha, Core-periphery structure in networks (revisited), SIAM Rev. 59 (3) (2017) 619–646.

[22] S. Kojaku, N. Masuda, Finding multiple core-periphery pairs in networks, Phys. Rev. E 96 (5) (2017) 052313.

[23] B.-B. Xiang, Z.-K. Bao, C. Ma, X. Zhang, H.-S. Chen, H.-F. Zhang, A unified method of detecting core-periphery structure and community structure in networks, Chaos 28 (1) (2018) 013122.

[24] D. Fasino, F. Rinaldi, A fast and exact greedy algorithm for the core-periphery problem, Symmetry 12 (2020) 94.

[25] S. Kojaku, N. Masuda, Core-periphery structure requires something else in the network, New J. Phys. 20 (4) (2018) 043012.

[26] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G.D. Melo, C. Gutierrez, S. Kirrane, J.E.L. Gayo, R. Navigli, S. Neumaier, et al., Knowledge graphs, ACM Comput. Surv. 54 (4) (2021) 1–37.

[27] B. Mandelbrot, Statistical self-similarity and fractional dimension, Science 156 (3775) (1967) 636–638.

[28] Y.-X. Kong, G.-Y. Shi, R.-J. Wu, Y. Zhang, K-core: Theories and applications, Phys. Rep. (2019).

[29] V. Chandola, A. Banerjee, V. Kumar, Anomaly detection: A survey, ACM Comput. Surv. 41 (3) (2009) 1–58.

[30] V. Hodge, J. Austin, A survey of outlier detection methodologies, Artif. Intell. Rev. 22 (2) (2004) 85–126.

[31] B.J. Stolz, J. Tanner, H.A. Harrington, V. Nanda, Geometric anomaly detection in data, Proc. Natl. Acad. Sci. 117 (33) (2020) 19664–19669.

[32] Kaggle Data-set, Covid-19 radiography database, 2020, https://www.kaggle.com/tawsifurrahman/covid19-radiography-database. Accessed: 2021-02-04.

Jianglong Yan received the B.S. degree in telecommunication engineering and M. Sc. degree in computer science, both from Zhengyuan University of Technology, China. Currently, he is a Ph.D. student at University of Sao Paulo, Brazil. His research interests are artificial neural networks, machine learning, complex networks, and pattern recognition.

Leandro Anghinoni received the BS degree in 2006 from the University of São Paulo in Production Engineering and the M.Sc. degree in 2018 from University of São Paulo in Applied Computing. He is currently a Ph.D. candidate in Computer Sciences and Computational Mathematics at University of São Paulo. His current research interests include Complex Networks, Machine Learning, Neural Networks and Graph Neural Networks.

Yu-tao Zhu received the B.S. degree in electronic engineering from Tsinghua University, China, in 2000, M.Sc. degree in communication engineering from the University of Melbourne, Australia, in 2003, and Ph.D. degree in communication engineering from Beijing University of Posts and Telecommunications, China, in 2016. He is currently the president of the China Branch of BRICS Institute of Future Network. He has been engaged in communication and information system research for a long time, his current research interests include 4G/5G mobile communication technology, 5G and NB IoT application research, artificial intelligence and industrial innovation.

Weiguang Liu received the B.S., and the M.Sc. and Ph.D. degrees from Xidian University, Xian, China, in 1988, 1994 and 2005, respectively, all in computer science. Currently, he is a full professor at Zhongyuan University of Technology, China. He has been a visiting scholar in Oklahoma State University, USA, from 2009 to 2010. He is in charge of the national first-class professional computer science and technology and the Natural Language Processing and Image Understanding of Zhengzhou Key Laboratory. His research interests are computer vision and intelligent systems.

Gen Li received the B.S. degree and M. Sc. degree, both in computer science, from Zhengyuan University of Technology, China. Currently, he is working at Shanghai Midu Information Technology Co. Ltd., China, as a researcher of algorithms. His research interests are artificial neural networks, machine learning, complex networks, and social network analysis.

Qiusheng Zheng received the B.S. degree from Shenyang University of Technology, China, in 1986 and M.Sc. degree from Zhengzhou University of Technology, China, in 1989. Currently, he is a full professor at Zhongyuan University of Technology, China. His research interests are social network analysis, artificial intelligence, and natural language processing.

Liang Zhao received the B.S. degree from Wuhan University, Wuhan, China, in 1988, and the M.Sc. and Ph.D. degrees from the Aeronautic Institute of Technology, Sao Jose dos Campos, Brazil, in 1996 and 1998, respectively, all in computer science. He joined the University of Sao Paulo (USP), Brazil, in 2000, where he is currently a Full Professor of the Department of Computing and Mathematics of the Faculty of Philosophy, Science, and Letters at Ribeirao Preto (FFCLRP) of USP. His current research interests include artificial neural networks, machine learning, complex networks, and pattern recognition. Dr. Zhao is a recipient of the Brazilian Research Productivity Fellowship. He was an Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS from 2009 to 2012. He is currently an Associate Editor of Neural Networks.

CHAPTER

5

TRANSGNN: A TRANSDUCTIVE GRAPH NEURAL NETWORK WITH GRAPH DYNAMIC EMBEDDING

Preprint, Leandro A., Yu-Tao Z., Donghong J. and Liang Z. "TransGNN: A Transductive Graph Neural Network with Graph Dynamic Embedding", accepted to IEEE International Joint Conference on Neural Networks, 2023.

TransGNN: A Transductive Graph Neural Network with Graph Dynamic Embedding

1st Leandro Anghinoni
Institute of Mathematics and Computer Sciences
University of São Paulo
São Carlos, Brazil
anghinoni@usp.br

2nd Yu-tao Zhu
China Branch of BRICS
Institute of Future Networks
ShenZhen, China
zhuyutao@bifnc.cn

3rd Donghong Ji
School of Cyber Science and Engineering
Wuhan University
Wuhan, China
dhji@whu.edu.cn

4th Liang Zhao
Faculty of Philosophy, Sciences and Letters
University of São Paulo
Ribeirão Preto, Brazil
zhao@usp.br

***Abstract*—Graph Neural Networks (GNNs) have become a rapidly growing field, due to their ability to capture the relationship among data, instead of only learning from the attribute of the data. The core of any GNN is the graph embedding generation by message passing mechanisms. In this work we propose a new message passing technique based on the Particle Competition and Cooperation (PCC) model, originally developed for community detection in graphs. The proposed framework performs a transductive learning in the network and passes the learned information to the nodes, prior to the inductive learning performed by traditional GNN schemes. The new GNN presents attractive features which overcomes the over-smoothing problem of traditional GNNs and shows promising results in terms of classification accuracy, computational cost and learning with very small quantity of labeled data.**

***Index Terms*—community detection, transductive learning, graph neural network, inductive learning**

I. Introduction

Over the past few years, Graph Neural Networks (GNNs) have become an important tool for modern problem solving and a hot topic of research. This framework is able to address several drawbacks of traditional neural networks, such as the assumption that the instances are independent of each other and the data representation in the Euclidean space. GNNs explicitly indicate the relation between any two instances and, by doing so, can be applied to any n^{th} dimensional space and to data that presents no spatial order. This characteristics also enhances the interpretability of the model, that tends to be lost in traditional neural network models during the convolution process. Several real world problems have been proven to be better solved in this paradigm, such as classification of citation networks and recommendation in social networks [1], [2], [3].

In general scheme of GNNs, a message passing function is used to pass information from a node to its neighbors and an aggregation function is used to update the node´s attribute at each iteration. This operation has to be permutation invariant, since the node has no physical position. The model learns an inductive function by propagating the error of each epoch, as in a neural network. One of the most relevant works in GNN is the Graph Convolutional Network (GCN) [3], which introduces a matrix normalization in the layer updating function to avoid the gradient exploding problem inherited from neural networks. It also paved the way for the construction of many other complex GNN models, including the most recent developments [1]. These end-to-end trainable models usually differ from each other by changing the pooling process, the readout process, the learning function or error propagation. Adding too many parameters, however, can make the model costly and too task specific. The time complexity of most recent models is $O(m)$, where m is the number of edges, since they usually operate on sparse matrices.

More recently, some works have proposed methods that are not trained end-to-end. In [4], for example, the authors combine label propagation and simple models to out-perform GNNs in certain cases. Some other works use mixed models, where part of learning process is done by traditional methods and part is done by GNNs. This can be done, for example, by performing random walks and learning which walks are more relevant for the classification of each specific node [5]. Other works have proposed adaptive propagation mechanisms by learning a homophily degree matrix prior to the propagation process [6].

Despite the advantages listed above, GNNs still presents several drawbacks and points to be explored. One of them is that adding too many layers to a GNN is known to lead to a over-smoothing of the graph [7], i.e., the local features of the data instances are lost, resulting in a shallow neural network. In the recent years, this problem has been dealt in several different ways, such as the application of the attention mechanisms to learn which neighbors and features are more important and by mixed models to capture local and global

This work was carried out at the Center for Artificial Intelligence (C4AI-USP), with support by the São Paulo Research Foundation (FAPESP grant #2019/07665-4) and by the IBM Corporation. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001.

features in different ways and combine it in the readout layer. Another salient problem is that the aggregation function must be permutation invariant, which, in general, only very simple functions (mean, max, etc) satisfy such a condition.

Complex networks are large-scale graphs with non-trivial connection patterns. One of the salient features of complex networks is the presence of communities, and community discovery in these systems has become a primary and prior object to help us understand how subgraphs interact and produce global behavior. From a topological structure standpoint, a community is a subgraph, in which the inner links are dense while the outer connections are relatively sparse. Community detection techniques have been around for decades. They are very good instruments to capture the structure of the graph and can learn the influence of one node over the other even without any label. Classifying data in the form of a graph is not recent and has been a field of research since the 70´s, when the first community detection algorithm was introduced [8]. These traditional methods mainly focus in the network topology and are usually regarded in the literature as community detection algorithms. These methods can be divided into seven different types: Graph Partition, Statistical Inference, Hierarchical Clustering, Dynamical Methods, Spectral Clustering, Density-based Algorithms and Optimizations [9]. Although the majority of the community detection algorithms works in an unsupervised manner, the Dynamical Methods can be used in the semi-supervised scheme by providing the labels of some nodes. These methods include WalkTrap [10], LPA [11], InfoMap [12] and Particle Competition and Cooperation (PCC) [13]. These methods are also based in transductive learning, where the solution is reached by reasoning over all the samples' attributes, instead of finding a general rule that can be generalized outside the observed samples.

In this work, we propose a new GNN model called 'TransGNN', in which the structure of the graph is learned prior to the inductive learning performed by the GNN and the information is embedded in the nodes. The proposed model takes advantage of the interpretability and low time complexity of the transductive PCC model as a message passing mechanism to capture the global structure of the data graph. Our study shows that the TransGNN, proposed in this work, leads to some advantages over a traditional GCN implementation and, therefore, we list the contributions of this work as follows:

- The method is easy to interpret because the Particle Competition and Cooperation heuristics in structure learning presents a natural inspired conquering-defend behavior.
- DeepGNN is still an open topic of research due to its inability to avoid over-smoothing when adding more than a few layers [7]. Our method is able to capture information from distant nodes without the over-smoothing problem, since the transductive learning is governed by independent particle walkings in the data graph with a pre-defined dynamics;
- Transductive learning, in general, can be performed with less labeled data and with less computational cost. Our method takes advantage of this characteristics to produce good results with a small portion of labeled data from the whole data set.

II. The Proposed Method

In this section, we present the proposed TransGNN in details. In the first sub-section, we introduce the necessary notations to describe the model; then, we give a short review on the PCC model, which will be used to learn the graph structure; after that, we show how the information obtained by the PPC model is incorporated to the original features to generate graph embedding. Finally, we describe the learning mechanism of the TransGNN. The proposed framework can be visualized in Figure 1.

A. Notation

A graph or network is defined as $G = (V, E)$, where $V = v_1, v_2, \ldots, v_n$ is the set of nodes and $E = e_{i,j}$ is the set of edges connecting v_i to v_j. $\boldsymbol{A}$ denotes the $n \times n$ adjacency matrix. If G is unweighted, then $a_{ij} = 1$ if nodes v_i and v_j are connected, otherwise $a_{ij} = 0$. If G is a weighted network, then $G = (V, E, W)$ and W is a weight matrix. In this paper we consider undirected graphs and each node is attributed, i.e., $v_i \in V$ is attributed by $\boldsymbol{x_i} \in \boldsymbol{X} \subseteq \mathbb{R}^{n \times d}$, or in the matrix form, $G = (V, E, X_1, X_2, \ldots, X_n)$.

B. Graph Structure Learning

The proposed method is based on the idea that the structural information of the graph can be learned before passing the node's embedding to a graph neural network. In this work, we use the Particle Competition and Cooperation (PCC) [13], [14] model as a message passing mechanism in TransGNN. Although we recommend the reader to refer to the original articles [13] and [14] for a deeper understanding of the method, we describe the main concepts of the technique here.

PCC is a graph-based semi-supervised learning technique. Given a data graph, each labeled node contains a particle and it walks in the graph based on a combined random-preferential rule. The particles of the same class (a team) cooperate among themselves, while the particles of different classes compete with each other to propagate class labels to the whole network. Finally, each team occupies a sub-graph corresponding to a data class. In this way, PCC propagates labels to all the unlabeled nodes (data items).

In PCC model, a dominance vector $v_i^{w_l}$ is attached to every node in the graph G. This vector has l positions, each one representing the dominance w of each class to a certain node i, therefore the summation over each vector is equal to one, as in Equation 1.

$$\sum_{l=1}^{L} v_i^{w_l} = 1 \tag{1}$$

where L is the total number of classes.

At the beginning, each dominance vector is started as shown by Equation 2, i.e., if the node is labeled, then the position of that class is set to 1 and the other positions to 0. If the node is

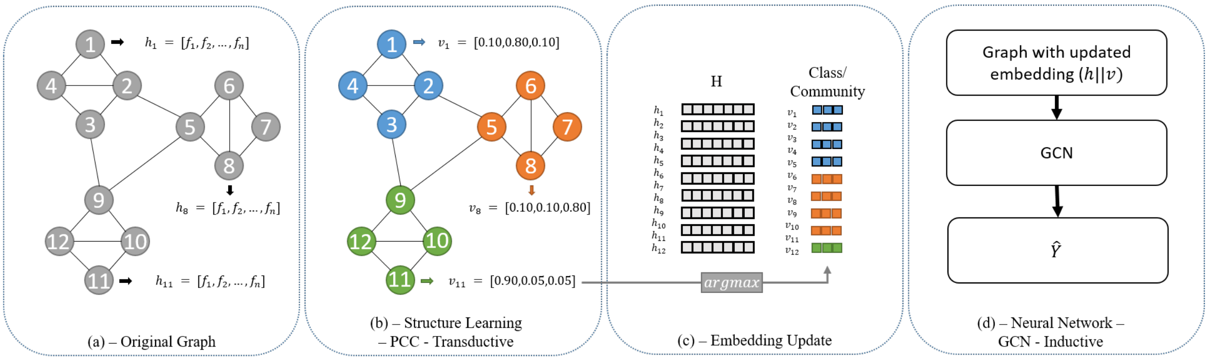

1
2
3
4
5
6
7
8
9
10
11
12
$h_1 = [f_1, f_2, \dots, f_n]$
$h_8 = [f_1, f_2, \dots, f_n]$
$h_{11} = [f_1, f_2, \dots, f_n]$
(a) – Original Graph
$v_1 = [0.10,0.80,0.10]$
$v_8 = [0.10,0.10,0.80]$
$v_{11} = [0.90,0.05,0.05]$
(b) – Structure Learning – PCC - Transductive
H
Class/ Community
h_1 … h_{12}
v_1 … v_{12}
argmax
(c) – Embedding Update
Graph with updated embedding $(h||v)$
GCN
$\hat{Y}$
(d) – Neural Network – GCN - Inductive

Fig. 1: Proposed framework. (a) Original Graph, where each node is represented by its original features h_n; (b) Structure learning performed by PCC, where each node receives a domination vector v_n, representing the probability of a node belonging to a certain community; (c) Embedding update by combining the original feature with the probable class of the node and (d) GCN learning over the updated embedding.

not labeled, then every position is set to the same dominance and equal to $1/l$.

$$v_i^{w_l}(0) = \begin{cases} 1 & \text{if} \quad y_i = l, \\ 0 & \text{if} \quad y_i \neq l \quad \text{and} \quad y_i \in L, \\ \frac{1}{l} & \text{if} \quad y_i = \emptyset. \end{cases} \tag{2}$$

In the semi-supervised scheme, the PCC method places a particle k in each labeled node. Once the algorithm starts, each particle visits a neighbor following Equation 3. The particle chooses a neighbor node to visit with the combination of preferential and random walk, represented by the probabilities P_{pref} and P_{rand}, respectively. A factor $\lambda \in [0,1]$ is used to balance the behavior of the particles between defensive and exploratory. This factor can be optimized for each graph G, depending on its topology.

$$P^{(k)}_{transition}(t) = \lambda P^{(k)}_{pref}(t) + (1-\lambda)P^{(k)}_{rand}(t) \tag{3}$$

The probabilities P_{rand} and P_{pref} are, respectively represented by Equations 4 and 5:

$$P^{(k)}_{rand}(i,j) = \frac{a_{i,j}}{\sum\limits_{u=1}^{V} a_{i,u}} \tag{4}$$

$$P^{(k)}_{pref}(i,j,t) = \frac{a_{i,j}\bar{N}^{(k)}_j(t)}{\sum\limits_{u=1}^{V} a_{i,u}\bar{N}^{(k)}_u(t)} \tag{5}$$

Notice that, in the random walk (representing an exploratory behavior), the probability of visiting a neighbor u is only proportional to the number of neighbors. The preferential walk (representing a defensive behavior), on the other hand, depends on the the vector $\bar{N}(t)$, which records the number of visits of each particle on each node and is defined as:

$$\bar{N}^{(k)}(t) = [\bar{N_1}^{(k)}(t), \bar{N_2}^{(k)}(t), \dots, \bar{N_V}^{(k)}(t)]^T \tag{6}$$

where k is the particle in consideration, $\bar{N}^{(k)}(t)$ records the number of visits of particle k on each node up to time t, and V is the total number of nodes in graph G.

The system dynamics can be summarized by the function ϕ (Equation 7), i.e., at each time step a particle in node i visits a neighbor j with probability $P^{(k)}_{transition}$, the vector $\bar{N}^{(k)}(t)$ is updated and the dominance vector is updated according to function γ.

$$\phi : \begin{cases} p^{(k)}(t+1) = j, \quad j \sim P^{(k)}_{transition}(t), \\ N^{(k)}_i(t+1) = N^{(k)}_i(t) + \mathbb{1}_{[p^{(k)}(t+1)=i]} \\ v_i^{w_l}(t+1) = \gamma \end{cases} \tag{7}$$

Every time a particle visits a node, it increases the dominance level of its team over that node and reduces the dominance level of other teams of particles on the same node. The dominance level of the node is updated as follows:

$$\gamma : v_i^{w_l}(t+1) = \begin{cases} \max(0, v_i^{w_l}(t) - \frac{\Delta_v \rho_j^w(t)}{L-1}) \\ \quad \text{if} \quad y_i = \emptyset \quad and \quad l \neq \rho_j^f, \\ v_i^{w_l}(t) + \sum\limits_{q \neq l} v_i^{w_q}(t) - v_i^{w_q}(t+1) \\ \quad \text{if} \quad y_i = \emptyset \quad and \quad l = \rho_j^f, \\ v_i^{w_l}(t) \quad \text{if} \quad y_i \in L \end{cases} \tag{8}$$

Conditions 1 and 2 are used when the particle visits an unlabeled node ($y_i = \emptyset$). Condition 1 rules how the dominance of opposite classes are decreased ($l \neq \rho_j^f$, where ρ_j^f is the particle class) and condition 2 rules how the dominance of the particle class is increased. Finally, if a particle visits a labeled node, the dominance vector is unchanged as in condition 3.

There are two other parameters in Equation 8. The first one is $\Delta_v \in [0,1] \subseteq \mathbb{R}$, that controls the rate of which the changes are made and is a settable parameter. The other one is ρ_v^w, which records the particle strength at time. The particle strength and its reanimation procedure are detailed in [13]. The algorithm stops when the number of iterations is reached or when the system reaches an equilibrium, i.e., the class of every node remains the same for a certain amount of iterations. The complete implementation is available at [15].

The equilibrium of the PCC system, however, is not guaranteed. In data-sets with low modularity, each run of the algorithm can present a significant difference in performance. This is common in real world applications where the clusters are sparse and sometimes overlapping. To overcome this situation we add a voting module after we perform a predefined number of runs. This voting module looks at the community assigned for each node individually across all the runs and assigns as the final community the most frequent label as described below:

$$v_i^{w_l} = Mo(v_i^{w_l}{}_{r_1}, v_i^{w_l}{}_{r_2}, \ldots, v_i^{w_l}{}_{r_n}) \tag{9}$$

where Mo is the mode function, $v_i^{w_l}$ is the dominance vector of a specific node and r_n is the number of the performed run out of n runs.

C. Embedding Update

At the end of the particle competition-cooperation process, each node has a dominance vector $v_i^{w_l}$ representing the probability that node i belongs to a certain community. In graphs with a high modularity, i.e., where the nodes are clustered in well-defined communities, it is common that $v_i^{w_l}$ ends up as a one hot encoding, with a few nodes (usually the high centrality ones) belonging to multiple communities. Our method, however, is focused in finding local relationships in the data, regardless of the global structure or the node's features. The intuition is that a node with a high probability of belonging to the same local structure will also present high feature similarity. By considering both information in the GNN we expect that the classification accuracy is improved since the local structure can act as decider for nodes with low feature similarity.

In this paper we use a concatenation rule, where the feature information h_i and the structure information $v_i^{w_l}$ are combined in the final feature $\hat{h_i}$ with different weights, defined by a parameter $\alpha \in [0, 1] \subseteq \mathbb{R}$. Equation 10 shows how each node is updated. Notice that we use a $argmax$ function in the vector $v_i^{w_l}$ to transform it into a one hot encoding.

$$\hat{h_i} = \Big\Vert_{i=1}^{N} (1-\alpha)h_i, \alpha(argmax(v_i^{w_l})) \tag{10}$$

As will be discussed in the Experimental Results, the parameter α can be adjusted to give more weight to the attribute vector or the structure vector. This will depend on the characteristics of the problem and the parameter can be optimized accordingly.

D. Classifier

In this paper we consider a two layer Graph Convolutional Network (GCN) with the same propagation rule as proposed in [3]:

$$\hat{H}_s^{(l+1)} = \sigma\left(D^{-\frac{1}{2}}AD^{-\frac{1}{2}}\hat{H}_s^{(l)}W_s^{(l)}\right) \tag{11}$$

where σ is the activation function (Relu), D is the graph degree matrix, A is the graph adjacency matrix and W is the layer-wise learnable weight matrix. $\hat{H}$ is the node embedding at the l^{th} layer and $\hat{H}(0)$ is equal to the updated embedding instead of the original features.

The predicted class $\hat{y}$ is obtained by applying $softmax$ to the the last layer, which has a size equal to the number of classes. We also use a standard cross-entropy as the loss function $\mathcal{L}$ to be minimized (Equation 12).

$$\begin{aligned} \hat{y}_i &= softmax(\hat{h}_i.W) \\ \mathcal{L} &= -\sum_{c=1}^{M} y_{i,c}\log(\hat{y}_{i,c}) \end{aligned} \tag{12}$$

E. Computational Complexity

Once the graph is constructed or in the case where the data is already in the form of a graph, PCC has linear time complexity, $O(n)$, if the graph is sparse and quadratic complexity, $O(n^2)$, otherwise, where n is the number of nodes [16]. Real world graphs frequently are sparse, therefore, we hope PCC works on linear time complexity em general.

GNNs, in general, have a time complexity of $O(m)$, where m is the number of edges. However, some models can be more costly, ranging from $O(n^2)$ as in DiffPool [17] to $O(n^3)$ as in PGC-DGCNN [18]. In this work we use a plain GCN as in [3], which has a time complexity of $O(m)$. Therefore our model has a time complexity of $O(n + m)$, i.e., linear to the sum of the number of the nodes and the number of the edges, for sparse graphs.

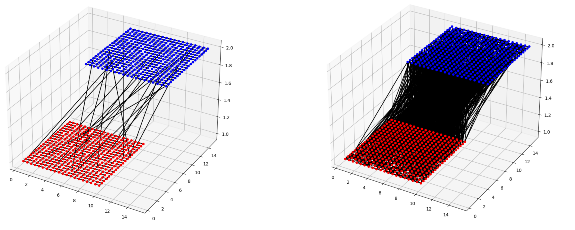
III. Method Application and Ablation Study

In this section we report the experimental results obtained by TransGNN. For this purpose, we use an artificial data-set and three well known public data-sets (Cora, Citeseer and Pubmed). Firstly, we explore the artificial toy data-set to clearly illustrate the advantages of the proposed model in different levels of class mixture and graph heterophily (neighboring nodes with different features). Then we apply TransGNN to the real data-sets and present an ablation study for some some of the real data-sets.

A. Artificial Toy Data-set

The toy data-set is composed of two classes of points forming two hyper-planes. The distance between every neighbor node of the same class is fixed. The two hyper-planes overlap and in the overlapped area the distance between two adjacent nodes from different classes is the same as the distance from an adjacent node of the same class. This area contains 25% of the total nodes. This setup is depicted in Figure 2.

The edges are added according to the following rule: for each class, the intra-class degree of every node is equal to three (d_{intra}=3), so each node is connected to 3 random neighbors of the same class. For the intra-class connections, two different



(a) Neighbor size = 1 and inter-class connections = 2%.

(b) Neighbor size = 2 and inter-class connections = 34%.

Fig. 2: Visualization of different setups of the artificial data. In (a) the nodes of the same class only connect to direct neighbors (1-hop) and the classes present little mix (2%). In (b) the nodes of the same class connect to nodes up to two steps away (2-hop) and the classes are very mixed (34%).

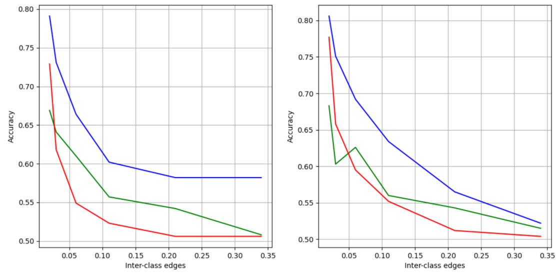


Fig. 3: Chart view of Table I showing the performance of the model under different scenarios in the artificial data-set. Left chart: 1-hop results; Right chart: 2-hop results.

scenarios were tested, considering the size of the neighborhood (1 and 2). Then, a number of inter-class edges are randomly added and in this case six scenarios are tested, depending on the amount of edges in relation to the total edges (2%, 3%, 6%, 11%, 21% and 34%). As for the labeled nodes, in every scenario we use 2% of the total nodes for training and 98% for testing. The training nodes are split in half for training and validation in the GCN.

In general, we can see from the results in Table I and Figure 3 that the particle competition performs better than the GCN when the inter-class edges rate is low, especially when the neighborhood size is larger. The GCN also has a very low performance when the neighborhood size is larger, which is expected since the network presents a high degree of heterophily in this case (nodes with different features are directly connected). In every scenario, however, the GCN is better than the PCC when the inter-class edges rate is high, since in this case the structure degrades but the GCN can still rely on the node´s feature. Finally, TransGNN can incorporate the advantages of both methods and keep a higher F1-Score in every scenario. In this experiment, we set $\alpha = 0.5$, giving the same weight to the features and the network structure. This way TransGNN can learn from both information.

TABLE I: Performance under different combinations of class mixture and feature dissimilarity. The amount of inter-class edges (mixture in the table) simulates the degree of mixture in the structure of different classes, while the the number of hops in the neighborhood of a node represents the dissimilarity allowed for two nodes to be connected.

	F1-score					
	1-hop			2-hop		
Mixture	GCN	PCC	TransGNN	GCN	PCC	TransGNN
2%	0.669	0.729	**0.791**	0.683	0.777	**0.806**
3%	0.641	0.618	**0.731**	0.603	0.658	**0.751**
6%	0.610	0.549	**0.664**	0.626	0.595	**0.692**
11%	0.557	0.523	**0.602**	0.560	0.552	**0.634**
21%	0.542	0.506	**0.582**	0.543	0.512	**0.565**
34%	0.508	0.506	**0.582**	0.515	0.504	**0.522**

B. Real Data-sets

All three data-sets (Cora, Citeseer and Pubmed) are citation data-sets. They are presented as graph objects, where each node represents a paper and a connection exists between two papers if either one cites the other. The node's features are bag-of-words vectors representing the most common words in the paper. The label of the node is the subject area of the paper.

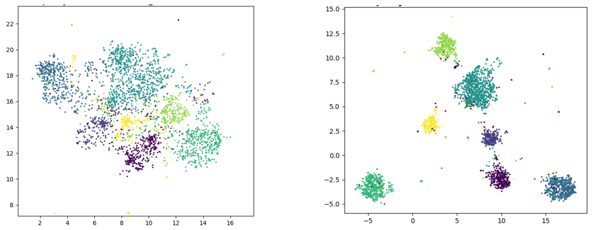
The data was downloaded from Spektral Python package [16] and the masks (training, validation and testing) provided by the package are used in the ablation study when no mask is specified.

The Cora data-set consists of 2708 nodes, 5429 edges and the nodes are divided into 7 classes. Each node contains a feature vector with 1433 positions (filled with either 0 or 1). The training mask contains 140 nodes, the validation mask 500 nodes and the test mask 1000 nodes. The Citeseer data-set consists of 3327 nodes, 9228 edges and the nodes are divided into 6 classes. Each node contains a feature vector with 3703 positions (filled with either 0 or 1). The training mask contains 120 nodes, the validation mask 500 nodes and the test mask 1000 nodes. The Pubmed data-set consists of 19717 nodes, 44338 edges and the nodes are divided into 3 classes. Each node contains a feature vector with 500 positions (filled with either 0 or 1). The training mask contains 60 nodes, the validation mask 500 nodes and the test mask 1000 nodes.

In order to visualize the problem and how the structure learning improves the data separation we perform a UMAP for supervised dimension reduction for Cora and Pubmed data-sets. The parameters used are: $n\_neighbors = 7$, $min\_dist = 0.5$ and $n\_components = 2$.

As we can see in Figure 4, in the original data, the classes are much closer than in the updated version, where the community information has been appended to the node embedding. This shows that this is step is able to learn the overall structure of the data, which will be useful in the inductive learning.

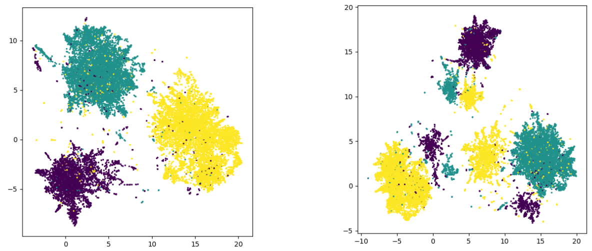
Now we compare the proposed method with recent state-of-the-art methods. The experiments setups were set to match



(a) Original

(b) Updated

Fig. 4: Umap of Cora features before (a) and after (b) embedding update.



(a) Pubmed Original

(b) Pubmed Updated

Fig. 5: Umap of Pubmed features before (a) and after (b) embedding update.

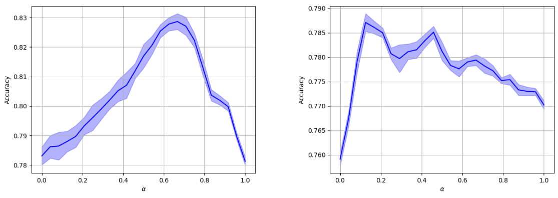
the ones used in [19] (such as the split rate and the number of times the algorithm is run), this way we present the same results with the addition of our model and the PCC when ran separately. We also use as the basic metrics the F1-Score and the MMC [19], since they are able to better express the performance of unbalanced data-sets.

As we can see in Table II, the proposed method presents a better performance in both Cora and Pubmed data-sets and stays in top three in the Citeseer data-set, ranking first in comparison with the other presented methods when considering the F1-Score.

C. Ablation Study

In this section we present an ablation study of the proposed method. We compare the proposed method with a pure PCC implementation and the classic GCN implementation as presented in [3]. We present this study because our method changes the attributes' representations of the nodes and not the learning architecture of the GNN, which means that our method could be generalized to most of the many GNN variations [1]. Therefore, our goal here is to demonstrate how our framework, which combines two different approaches, performs when compared to its related parts. In this part we use Cora and Pubmed data-sets.

The first experiment was conducted to evaluate the effect of α, as introduced in Equation 10, in the final accuracy. To do so, vary α in the range $[0, 1]$ with a step of $\frac{1}{26}$. When $\alpha = 0$, the model gives no weight to the PCC structure and only considers the attributes of the nodes. On the other hand, when $\alpha = 1$ the model only considers the graph structure and ignores the attributes of the nodes by the natural of PCC model.



(a) Alpha Cora

(b) Alpha Pubmed

Fig. 6: Classification accuracy for different values of alpha on Cora and Pubmed data-sets.

For both the Cora and Pubmed Data-sets, we see that the best classification accuracy is reached when we combine both information (Figure 6). The Cora data-set reaches its peak accuracy with $\alpha = 0.666$, while the Pubmed data-set reaches its peak accuracy tieh $\alpha = 0.125$. As we can see, α must be set according to the data-set, since each one might present different properties. While the Cora data-set depends more on the structure of the data, the Pubmed data-set depends more on the attributes of the nodes.

In Figure 7, we plot the boxplot of the three methods for the Cora data-set (a) and for the Pubmed data-set (b). In this simulations, TransGNN uses the enhanced attribute $\hat{h}$, while the GCN uses the original attribute h. The PCC method only uses the labels and the adjacency matrix by its nature.

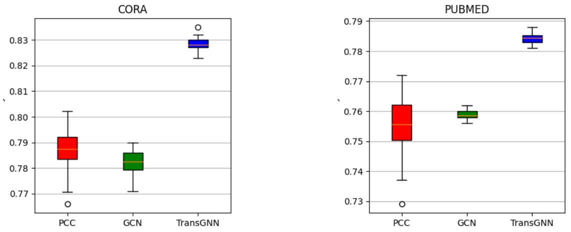
For the Cora data-set we can see that the PCC method performs better than a plain GCN ($acc\_pcc = 0.787, acc\_gcn = 0.782$), however PCC is not stable, with a high standard deviation and a high spread from the lowest to highest accuracy ($std\_pcc = 0.007, std\_gcn = 0.004$). TransGNN presents a better stability than the GCN ($std\_transgnn = 0.002$) with a significant higher accuracy than the plain GCN ($acc\_transgnn = 0.829$).

As for the Pubmed data-set, the GCN presents a slightly higher accuracy than the PCC ($acc\_pcc = 0.756, acc\_gcn = 0.759$), with the PCC with a higher standard deviation as before ($std\_pcc = 0.008, std\_gcn = 0.002$). However, TransGNN again performs significantly better in this data-set and presents a very low standard deviation ($acc\_transgnn = 0.784, std\_transgnn = 0.002$). All of these results are in line with the previous results, that showed that the structure information is more relevant to the Cora data-set.

Another feature of the proposed model is that it outperforms the compared methods with any number of labeled samples. In Figure 8 we show the accuracy of the model for the Cora Data-set depending on how many samples are labeled per class. With one labeled sample per class, the proposed model matches the performance of the PCC model and outperforms by a lot the plain GCN. In the range 1 to 10 samples, the

TABLE II: Comparison of proposed method against state-of-the-art methods. Results from methods 1-9 (above the break line) are replicated from [19]. Results in bold are the best results for the data-set, while the underlined results are the other two top three results. Both the F1-Score and MMC are reported for each data-set. The rank is based on the F1-Score, that is more commonly used in other works.

	CORA		CITESEER		PUBMED		
	F1-Score	MMC	F1-Score	MMC	F1-Score	MMC	Rank
ChebNet	0.6551±0.0115	0.5367±0.0087	0.5767±0.0124	0.5196±0.0115	0.6874±0.0072	0.5761±0.0094	10.67
GraphSAGE	0.6848±0.0071	0.5823±0.0058	0.6014±0.0097	0.5474±0.0061	0.7349±0.0043	0.6043±0.0047	9.67
GCN	0.6861±0.0023	0.6146±0.0014	0.6158±0.0029	0.5549±0.0014	0.7524±0.0023	0.6265±0.0038	8.33
GAT	0.7134±0.0072	0.6379±0.0061	0.6290±0.0085	0.5662±0.0107	0.7516±0.0034	0.6234±0.0028	7.33
Grand	0.7156±0.0059	0.6484±0.0045	0.6248±0.0057	0.5838±0.0046	0.7614±0.0053	0.6357±0.0061	6.33
GCNII	0.7162±0.0064	0.6531±0.0076	0.6235±0.0078	0.5861±0.0069	0.7586±0.0047	0.6376±0.0052	6.67
GraphSMOTE	0.7213±0.0075	0.6553±0.0066	0.6294±0.0091	0.6113±0.0083	0.7649±0.0045	0.6399±0.0047	4.33
DR-GCN	0.7247±0.0057	0.6588±0.0065	0.6332±0.0049	0.6143±0.0038	0.7659±0.0043	0.6428±0.0046	3.33
GNN-INCM	0.7508±0.0045	0.7237±0.0051	**0.6490±0.0048**	**0.6274±0.0036**	0.7704±0.0039	0.6493±0.0059	2.00
PCC	0.7170±0.0100	0.6700±0.0120	0.5380±0.0120	0.4450±0.0140	0.7860±0.0030	0.6790±0.0040	6.00
TransGNN	**0.7730±0.0180**	**0.7350±0.0210**	0.6340±0.0160	0.5600±0.0190	**0.8110±0.0080**	**0.7170±0.0110**	**1.33**



(a) Boxplot Core

(b) Boxplot Pubmed

Fig. 7: Accuracy comparison of the proposed method against its parts.

proposed method outperforms both methods and than it starts to show a similar performance as the GCN. In this study, the labeled samples are randomly chosen and are different than the ones provided by the Spektral Data-set, therefore the accuracy with the same number of labeled samples is not comparable. Also, we test the model in all the samples not used as the training set.

IV. Conclusion

In this work we have proposed a GNN that uses dynamic variable values of PCC as message passing mechanism to enhance the nodes attributes. We showed that by learning the network structure in advance and adding this information to the feature matrix, a plain GCN can learn better than using only the original attributes. As we have showed, the weight of each element (network structure / node attribute) can be optimized in a way that the performance of the mixed model is better than its parts. Also, the proposed model has shown promising results when compared to other state-of-the-art methods.

Another important feature of the the method is its ability to incorporate information from distant nodes to the nodes' embedding without over-smoothing the data graph, which is still a problem to be solved in deep graph neural network structures.

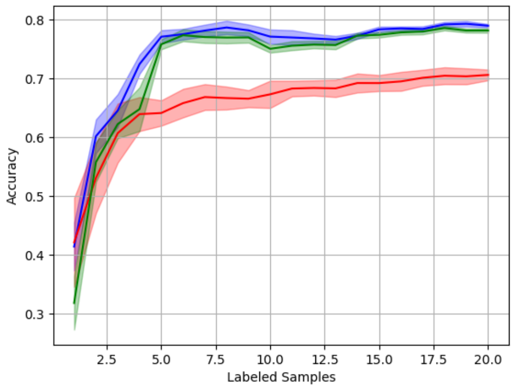


Fig. 8: Influence of training set size in the Cora Data-set accuracy. In blue the proposed method accuracy (TransGNN), in green the plain GCN accuracy and in red the PCC accuracy.

As a future work, we believe more applications can be explored, such as image segmentation and classification, where the structure of image tiles can be as significant as the tiles attributes. Another field to be explored is the combination of different transductive/inductive methods. Nowadays there as several transduction methods with high precision and low complexity and several GNN variations which are task specific. We believe that combinations like the one we have proposed may help improving this field of research.

References

[1] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Philip, S. Y. (2020). A comprehensive survey on graph neural networks. IEEE transactions on neural networks and learning systems, 32(1), 4-24.

[2] Chatzianastasis, M., Lutzeyer, J. F., Dasoulas, G., & Vazirgiannis, M. (2022). Graph Ordering Attention Networks. arXiv preprint arXiv:2204.05351.

[3] Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.

[4] Huang, Q., He, H., Singh, A., Lim, S. N., & Benson, A. R. (2020). Combining label propagation and simple models out-performs graph neural networks. arXiv preprint arXiv:2010.13993.

[5] Jin, D., Wang, R., Ge, M., He, D., Li, X., Lin, W., & Zhang, W. (2022). Raw-gnn: Random walk aggregation based graph neural network. arXiv preprint arXiv:2206.13953.

[6] Wang, T., Jin, D., Wang, R., He, D., Huang, Y. (2022, June). Powerful graph convolutional networks with adaptive propagation mechanism for homophily and heterophily. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 36, No. 4, pp. 4210-4218).

[7] Oono, K., & Suzuki, T. (2019). Graph neural networks exponentially lose expressive power for node classification. arXiv preprint arXiv:1905.10947.

[8] Kernighan, B. W., & Lin, S. (1970). An efficient heuristic procedure for partitioning graphs. The Bell system technical journal, 49(2), 291-307.

[9] Su, X., Xue, S., Liu, F., Wu, J., Yang, J., Zhou, C., ... & Philip, S. Y. (2022). A comprehensive survey on community detection with deep learning. IEEE Transactions on Neural Networks and Learning Systems.

[10] Pons, P., & Latapy, M. (2005). Computing communities in large networks using random walks. In Computer and Information Sciences-ISCIS 2005: 20th International Symposium, Istanbul, Turkey, October 26-28, 2005. Proceedings 20 (pp. 284-293). Springer Berlin Heidelberg.

[11] Rosvall, M., & Bergstrom, C. T. (2008). Maps of random walks on complex networks reveal community structure. Proceedings of the national academy of sciences, 105(4), 1118-1123.

[12] Raghavan, U. N., Albert, R., & Kumara, S. (2007). Near linear time algorithm to detect community structures in large-scale networks. Physical review E, 76(3), 036106.

[13] Breve, F., Zhao, L., Quiles, M., Pedrycz, W., & Liu, J. (2011). Particle competition and cooperation in networks for semi-supervised learning. IEEE Transactions on Knowledge and Data Engineering, 24(9), 1686-1698.

[14] Silva, T. C., & Zhao, L. (2012). Stochastic competitive learning in complex networks. IEEE Transactions on Neural Networks and Learning Systems, 23(3), 385-398.

[15] Implementation - PCC. Retrieved November 8, 2022, fromhttps://github.com/fbreve/Particle-Competition-and-Cooperation.

[16] Datasets - Spektral. Datasets - Spektral. Retrieved November 8, 2022, from https://graphneural.network/datasets/.

[17] Ying, Z., You, J., Morris, C., Ren, X., Hamilton, W., & Leskovec, J. (2018). Hierarchical graph representation learning with differentiable pooling. Advances in neural information processing systems, 31.

[18] Tran, D. V., Navarin, N., & Sperduti, A. (2018, November). On filter size in graph convolutional networks. In 2018 ieee symposium series on computational intelligence (ssci) (pp. 1534-1541). IEEE.

[19] Huang, Z., Tang, Y., & Chen, Y. (2022). A graph neural network-based node classification model on class-imbalanced graph data. Knowledge-Based Systems, 244, 108538.

CHAPTER 6

CONCLUSION

Machine learning models based on the structure of complex networks can be defined in a general framework composed of three steps. Firstly, the vector-based data, such as times series and images, must be converted to a network. This first step encompasses the network construction. In the second step, the network structure is characterized, using the training data to split the labeled samples into separated structures, such as clusters, core structures and periphery structures. Finally, new instances are classified according to their similarities to the training data or via models that process the whole graph with the learned information. In this research, we have studied problems with different network structures and proposed particular solutions to each of the learning tasks upon the networks. The contributions include not only the results presented in the previous Chapters but also the understanding of local characteristics versus global characteristics and how the models could deal with these differences in novel ways.

Besides of the presented papers, the author of this thesis has also made important contributions to the following paper within the same research topic: "Gao, X., Zheng, Q., Vega-Oliveros, D. A., Anghinoni, L., & Zhao, L. (2020). Temporal network pattern identification by community modelling. Scientific Reports, 10(1), 240. <https://doi.org/10.1038/s41598-019-57123-1>." The paper is not included in the thesis because he is the third author.

6.1 Concluding remarks

In this section, each topic of research is commented and the concluding remarks are listed below.

In Chapter 2, it is presented a paper that transforms a time series into a complex network considering the time series as a sequence of phase space vectors, which are then connected following a threshold in the Pearson correlation of these segments. The final product of this step is a recurrence network, where each pattern of a temporal segment is a node and it is connected to the next temporal pattern. The study shows that these temporal segment patterns tend to form clusters, meaning that some patterns are more likely to be connected to each other. This is an interesting observation, since such a phenomenon is not observed in the original stochastic time series. When the network is mapped back to the time series, the study shows that even stochastic series present recurrent temporal sequences, which may be useful for prediction purposes.

In Chapter 3, a study was conducted to predict the spreading rate in the early stages of the Covid-19 outbreak in Brazil. It is worth noticing that this research was conducted in the beginning of the outbreak, when the data started to become public. Still, the research was able to present some insightful results, regarding the effects of public measures and predicting the peak date and the percentage number of infected people at this date for every available city in Brazil.

In Chapter 4, the data patterns are characterized by a different type o structure, the core-periphery network. In this structure, the core data are very well connected while the periphery data is sparse. This research has shown that the core-periphery structure can be useful to represent certain types of data organizations, specifically when one of the classes has a high dispersion. To evaluate this, we construct a network from the x-ray chest images, where the core represents the normal lung class and the periphery represents the Covid-19 class with dispersed feature The model is able to train a network by optimizing the core-periphery structure in a way the new unlabeled instances could be classified accordingly.

Finally, in Chapter 5, the community structure of networks is used again to enhance the node embedding of a GNN architecture. In this case, the proposed model is not trained end-to-end but, instead, is composed of two main steps: (i) the structure learning, which is achieved by a transductive method and (ii) the classification process, which is done by an inductive method. The paper shows that, if the network presents some degree of clustering, i.e., higher modularity than random, then the GNN should benefit from the information acquired at the first step.

6.2 Future works

Although the presented researches have tackled specific problems, some future works can be listed based on these contributions.

1. Studying how overlapping communities interferes in the hierarchical community representation of the time series.
2. Modeling stochastic time series via Markov Chain process built from clustered networks.
3. Development of predictive models based on LSTM neural networks and the pattern sequence generated from the clustered network.
4. Revisiting the network SIR model to incorporate recent pandemic data and proposing a new model for future events.

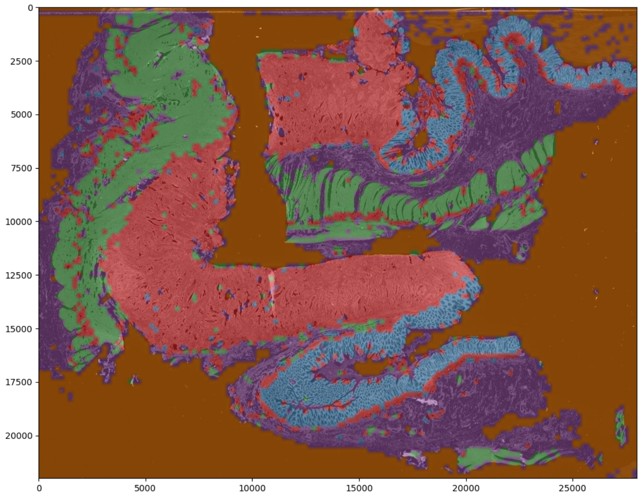


Figure 3 – Preliminary result of the novel TransGNN with attention. The whole slide image of a gastrointestinal sample is segmented into five different classes, including tumor and normal regions. Visual comparison with annotated slides suggests a good performance of the model.

5. Proposal for expansion of the core-periphery model to multi-core periphery model, in order to capture different dispersion levels and multi-class problems.

6. Incorporating attention mechanisms to the TransGNN model. This can be achieved from the information generated in the particle competition step by generating weights to the edges of the graph based on dominance similarity. This topic of research is an extension of the work developed in this thesis and is currently under development with some good preliminary results. We have applied this novel TransGNN in the field of digital pathology, specifically to segment H&E-stained whole slide images, as in Figure 3. This new study should be finished in a short period of time.

7. Proposal for a deep GNN model based on the network structure. A possible way of addressing this issue can be performing the message passing hierarchically on different community levels, avoiding the over-smoothing problem.

BIBLIOGRAPHY

BAHETI, A.; TOSHNIWAL, D. Trend analysis of time series data using data mining techniques. In: IEEE. **2014 IEEE International Congress on Big Data**. [S.l.], 2014. p. 430–437. Citation on page 21.

BARABÁSI, A.-L. Network science. **Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences**, The Royal Society Publishing, v. 371, n. 1987, p. 20120375, 2013. Citation on page 16.

BISHOP, C. M.; NASRABADI, N. M. **Pattern recognition and machine learning**. [S.l.]: Springer, 2006. Citations on pages 15, 16, and 21.

BOCCALETTI, S.; LATORA, V.; MORENO, Y.; CHAVEZ, M.; HWANG, D.-U. Complex networks: Structure and dynamics. **Physics reports**, Elsevier, v. 424, n. 4-5, p. 175–308, 2006. Citation on page 16.

BORGATTI, S. P.; EVERETT, M. G. Models of core/periphery structures. **Social networks**, Elsevier, v. 21, n. 4, p. 375–395, 2000. Citation on page 16.

BREVE, F.; ZHAO, L.; QUILES, M.; PEDRYCZ, W.; LIU, J. Particle competition and cooperation in networks for semi-supervised learning. **IEEE Transactions on Knowledge and Data Engineering**, IEEE, v. 24, n. 9, p. 1686–1698, 2011. Citation on page 18.

BUNKE, H.; RIESEN, K. Improving vector space embedding of graphs through feature selection algorithms. **Pattern Recognition**, Elsevier, v. 44, n. 9, p. 1928–1940, 2011. Citation on page 18.

CARNEIRO, M. G.; CHENG, R.; ZHAO, L.; JIN, Y. Particle swarm optimization for network-based data classification. **Neural Networks**, Elsevier, v. 110, p. 243–255, 2019. Citation on page 18.

CARNEIRO, M. G.; ZHAO, L. Organizational data classification based on the importance concept of complex networks. **IEEE transactions on neural networks and learning systems**, IEEE, v. 29, n. 8, p. 3361–3373, 2017. Citations on pages 16 and 18.

CHAPELLE, O.; SCHOLKOPF, B.; ZIEN, A. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. **IEEE Transactions on Neural Networks**, IEEE, v. 20, n. 3, p. 542–542, 2009. Citation on page 18.

CHEN, D.; LV, J.; YI, Z. Graph regularized restricted boltzmann machine. **IEEE transactions on neural networks and learning systems**, IEEE, v. 29, n. 6, p. 2651–2659, 2017. Citation on page 18.

COLLIRI, T.; JI, D.; PAN, H.; ZHAO, L. A network-based high level data classification technique. In: IEEE. **2018 International Joint Conference on Neural Networks (IJCNN)**. [S.l.], 2018. p. 1–8. Citation on page 16.

DANON, L.; DIAZ-GUILERA, A.; DUCH, J.; ARENAS, A. Comparing community structure identification. **Journal of statistical mechanics: Theory and experiment**, IOP Publishing, v. 2005, n. 09, p. P09008, 2005. Citation on page 16.

FORTUNATO, S. Community detection in graphs. **Physics reports**, Elsevier, v. 486, n. 3-5, p. 75–174, 2010. Citations on pages 16 and 18.

GONG, C.; LIU, T.; TAO, D.; FU, K.; TU, E.; YANG, J. Deformed graph laplacian for semisupervised learning. **IEEE transactions on neural networks and learning systems**, IEEE, v. 26, n. 10, p. 2261–2274, 2015. Citation on page 18.

HINTON, G.; DENG, L.; YU, D.; DAHL, G. E.; MOHAMED, A.-r.; JAITLY, N.; SENIOR, A.; VANHOUCKE, V.; NGUYEN, P.; SAINATH, T. N. *et al.* Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. **IEEE Signal processing magazine**, IEEE, v. 29, n. 6, p. 82–97, 2012. Citation on page 18.

IOSIFIDIS, A.; TEFAS, A.; PITAS, I. Graph embedded extreme learning machine. **IEEE transactions on cybernetics**, IEEE, v. 46, n. 1, p. 311–324, 2015. Citation on page 17.

KEELING, M. J.; EAMES, K. T. Networks and epidemic models. **Journal of the royal society interface**, The Royal Society London, v. 2, n. 4, p. 295–307, 2005. Citation on page 21.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **nature**, Nature Publishing Group UK London, v. 521, n. 7553, p. 436–444, 2015. Citations on pages 15, 16, and 21.

LIU, T.; HUANG, J.; HE, Z.; ZHANG, Y.; YAN, N.; ZHANG, C. J.; MING, W.-K. A real-world data validation of the value of early-stage sir modelling to public health. **Scientific Reports**, Nature Publishing Group UK London, v. 13, n. 1, p. 9164, 2023. Citation on page 21.

LUONG, M.-T.; PHAM, H.; MANNING, C. D. Effective approaches to attention-based neural machine translation. **arXiv preprint arXiv:1508.04025**, 2015. Citation on page 18.

MITCHELL, T. M. *et al.* **Machine learning**. [S.l.]: McGraw-hill New York, 2007. Citation on page 15.

NEWMAN, M. E.; GIRVAN, M. Finding and evaluating community structure in networks. **Physical review E**, APS, v. 69, n. 2, p. 026113, 2004. Citation on page 16.

NI, B.; YAN, S.; KASSIM, A. Learning a propagable graph for semisupervised learning: Classification and regression. **IEEE Transactions on Knowledge and Data Engineering**, IEEE, v. 24, n. 1, p. 114–126, 2010. Citation on page 18.

NICKEL, M.; MURPHY, K.; TRESP, V.; GABRILOVICH, E. A review of relational machine learning for knowledge graphs. **Proceedings of the IEEE**, IEEE, v. 104, n. 1, p. 11–33, 2015. Citation on page 17.

OONO, K.; SUZUKI, T. Graph neural networks exponentially lose expressive power for node classification. **arXiv preprint arXiv:1905.10947**, 2019. Citation on page 22.

RANI, S. *et al.* Review on time series databases and recent research trends in time series mining. In: IEEE. **2014 5th International Conference-Confluence The Next Generation Information Technology Summit (Confluence)**. [S.l.], 2014. p. 109–115. Citation on page 21.

REDMON, J.; DIVVALA, S.; GIRSHICK, R.; FARHADI, A. You only look once: Unified, real-time object detection. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2016. p. 779–788. Citation on page 18.

REN, S.; HE, K.; GIRSHICK, R.; SUN, J. Faster r-cnn: Towards real-time object detection with region proposal networks. **Advances in neural information processing systems**, v. 28, 2015. Citation on page 18.

RIESEN, K.; BUNKE, H. Reducing the dimensionality of dissimilarity space embedding graph kernels. **Engineering Applications of Artificial Intelligence**, Elsevier, v. 22, n. 1, p. 48–56, 2009. Citation on page 18.

SARKER, I. H. Machine learning: Algorithms, real-world applications and research directions. **SN computer science**, Springer, v. 2, n. 3, p. 160, 2021. Citation on page 15.

SIAMI-NAMINI, S.; TAVAKOLI, N.; NAMIN, A. S. A comparison of arima and lstm in forecasting time series. In: IEEE. **2018 17th IEEE international conference on machine learning and applications (ICMLA)**. [S.l.], 2018. p. 1394–1401. Citation on page 21.

SILVA, T. C.; ZHAO, L. Network-based high level data classification. **IEEE Transactions on Neural Networks and Learning Systems**, IEEE, v. 23, n. 6, p. 954–970, 2012. Citations on pages 16 and 18.

____. Stochastic competitive learning in complex networks. **IEEE Transactions on Neural Networks and Learning Systems**, IEEE, v. 23, n. 3, p. 385–398, 2012. Citation on page 18.

____. High-level pattern-based classification via tourist walks in networks. **Information Sciences**, Elsevier, v. 294, p. 109–126, 2015. Citations on pages 16 and 18.

____. **Machine learning in complex networks**. [S.l.]: Springer, 2016. Citation on page 18.

STEGEHUIS, C.; HOFSTAD, R. V. D.; LEEUWAARDEN, J. S. V. Epidemic spreading on complex networks with community structures. **Scientific reports**, Springer, v. 6, n. 1, p. 1–7, 2016. Citation on page 21.

STROGATZ, S. H. Exploring complex networks. **nature**, Nature Publishing Group UK London, v. 410, n. 6825, p. 268–276, 2001. Citation on page 16.

VERRI, F. A. N.; URIO, P. R.; ZHAO, L. Network unfolding map by vertex-edge dynamics modeling. **IEEE Transactions on Neural Networks and Learning Systems**, IEEE, v. 29, n. 2, p. 405–418, 2016. Citation on page 18.

WARD, I. R.; JOYNER, J.; LICKFOLD, C.; GUO, Y.; BENNAMOUN, M. A practical tutorial on graph neural networks. **ACM Computing Surveys (CSUR)**, ACM New York, NY, v. 54, n. 10s, p. 1–35, 2022. Citation on page 19.

WU, Y.; SCHUSTER, M.; CHEN, Z.; LE, Q. V.; NOROUZI, M.; MACHEREY, W.; KRIKUN, M.; CAO, Y.; GAO, Q.; MACHEREY, K. *et al.* Google's neural machine translation system: Bridging the gap between human and machine translation. **arXiv preprint arXiv:1609.08144**, 2016. Citation on page 18.

WU, Z.; PAN, S.; CHEN, F.; LONG, G.; ZHANG, C.; PHILIP, S. Y. A comprehensive survey on graph neural networks. **IEEE transactions on neural networks and learning systems**, IEEE, v. 32, n. 1, p. 4–24, 2020. Citations on pages 19 and 22.

ZHANG, J.; WANG, Y.; MOLINO, P.; LI, L.; EBERT, D. S. Manifold: A model-agnostic framework for interpretation and diagnosis of machine learning models. **IEEE transactions on visualization and computer graphics**, IEEE, v. 25, n. 1, p. 364–373, 2018. Citation on page 18.

ZHANG, K.; LAN, L.; KWOK, J. T.; VUCETIC, S.; PARVIN, B. Scaling up graph-based semisupervised learning via prototype vector machines. **IEEE transactions on neural networks and learning systems**, IEEE, v. 26, n. 3, p. 444–457, 2014. Citation on page 18.

ZHANG, Q.; SONG, X.; SHAO, X.; ZHAO, H.; SHIBASAKI, R. Object discovery: Soft attributed graph mining. **IEEE transactions on pattern analysis and machine intelligence**, IEEE, v. 38, n. 3, p. 532–545, 2015. Citation on page 18.

ZHANG, Q.-s.; ZHU, S.-C. Visual interpretability for deep learning: a survey. **Frontiers of Information Technology & Electronic Engineering**, Springer, v. 19, n. 1, p. 27–39, 2018. Citation on page 21.

ZHANG, Z.; CUI, P.; ZHU, W. Deep learning on graphs: A survey. **IEEE Transactions on Knowledge and Data Engineering**, IEEE, v. 34, n. 1, p. 249–270, 2020. Citations on pages 17 and 19.

ZHOU, J.; CUI, G.; HU, S.; ZHANG, Z.; YANG, C.; LIU, Z.; WANG, L.; LI, C.; SUN, M. Graph neural networks: A review of methods and applications. **AI open**, Elsevier, v. 1, p. 57–81, 2020. Citation on page 19.

ZHOU, Z.-H. **Machine learning**. [S.l.]: Springer Nature, 2021. Citations on pages 15 and 16.

ZHU, X. J. Semi-supervised learning literature survey. University of Wisconsin-Madison Department of Computer Sciences, 2005. Citation on page 18.

ICMC USP
SÃO CARLOS