UNIVERSIDADE DE SÃO PAULO

Instituto de Ciências Matemáticas e de Computação

Time series data mining using complex networks

Leonardo Nascimento Ferreira

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

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Leonardo Nascimento Ferreira

Time series data mining using complex networks

Doctoral dissertation submitted to the Institute of Mathematics and Computer Sciences – 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

USP – São Carlos
November 2017

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

N383t Nascimento Ferreira, Leonardo
Time series data mining using complex networks /
Leonardo Nascimento Ferreira; orientador Zhao
Liang. -- São Carlos, 2017.
147 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, 2017.

1. Time series. 2. Data mining. 3. Complex
networks. 4. Classification. 5. Clustering. I.
Liang, Zhao, orient. II. Título.

Leonardo Nascimento Ferreira

Mineração de dados em séries temporais usando redes complexas

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

USP – São Carlos
Novembro de 2017

I dedicate this PhD thesis to my parents, Ludgero and Silvana.

ACKNOWLEDGEMENTS

First, I would like to thank my advisor Prof. Zhao Liang. I will never forget the first time that I knocked the door of his office. I was starting the master's degree, and I was very anxious about what a graduation would be. He calmly explained it to me and helped me through this path. During my Ph.D., he moved to another campus. We kept working distant but he was always there when I needed. Thanks for trusting me and for letting me free to pursue my research interests. Thanks for teaching me so many things.

I want to thank Prof. Luis Amaral, my "second" advisor. Thanks for receiving me at Northwestern University. During this visit, I learned many research skills, but I also discovered how important non-technical skills are. I admire his friendship with the other students and the way he conducts his lab. I have learned a lot from him and all the other students and researchers at the Amaral-Lab. I truly returned to Brazil as a better researcher.

I thank Prof. Eamonn Keogh from the University of California at Riverside for providing the dataset used in the experiments. Without all these datasets, this thesis would not be possible.

I thank the university staff for helping me so many times on many different issues. During my Ph.D., I lived in different cities, and they always helped me by email or phone. It saved me time and money.

I would like to thank the financial support from CNPq and CAPES. The "Science without Borders" is a fantastic program. Through this scholarship, I could have a fascinating and enriching experience in one of the best universities in the world. I promise to return this investment by being a better professional.

My family has a significant influence in my life as researcher and human being. Thanks for always encouraging my studies. Thanks for supporting my "crazy" idea of quitting a good job to dedicate just for my Ph.D. They quickly understood that I love the university place and that is the place that I want to be. Thanks also to my brother Lucas and my sister Lorrana for the good times together.

I also want to thank my wife, friend, and sweetie Nicole. She knows exactly how difficult it is to become a Ph.D. and always motivated me. Thanks for reviewing this thesis and all the contributions. Thanks for being by my side for all these years.

Finally, I want to thank all the friends, colleges and familiars that help me directly or indirectly with this thesis.

*“If data had mass,
the earth would be a black hole.”
(Stephen Marsland)*

RESUMO

FERREIRA, L. N. **Mineração de dados em séries temporais usando redes complexas**. 2017. 147 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, 2017.

Séries temporais são conjuntos de dados ordenados no tempo. Devido à ubiquidade desses dados, seu estudo é interessante para muitos campos da ciência. A mineração de dados temporais é uma área de pesquisa que tem como objetivo extrair informações desses dados relacionados no tempo. Para isso, modelos são usados para descrever as séries e buscar por padrões. Uma forma de modelar séries temporais é por meio de redes complexas. Nessa modelagem, um mapeamento é feito do espaço temporal para o espaço topológico, o que permite avaliar dados temporais usando técnicas de redes. Nesta tese, apresentamos soluções para tarefas de mineração de dados de séries temporais usando redes complexas. O objetivo principal foi avaliar os benefícios do uso da teoria de redes para extrair informações de dados temporais. Concentramo-nos em três tarefas de mineração. (1) Na tarefa de agrupamento, cada série temporal é representada por um vértice e as arestas são criadas entre as séries de acordo com sua similaridade. Os algoritmos de detecção de comunidades podem ser usados para agrupar séries semelhantes. Os resultados mostram que esta abordagem apresenta melhores resultados do que os resultados de agrupamento tradicional. (2) Na tarefa de classificação, cada série temporal rotulada em um banco de dados é mapeada para um gráfico de visibilidade. A classificação é realizada transformando uma série temporal não marcada em um gráfico de visibilidade e comparando-a com os gráficos rotulados usando uma função de distância. O novo rótulo é dado pelo rótulo mais frequente nos k grafos mais próximos. (3) Na tarefa de detecção de periodicidade, uma série temporal é primeiramente transformada em um gráfico de visibilidade. Máximos locais em uma série temporal geralmente são mapeados para vértices altamente conectados que ligam duas comunidades. O método proposto utiliza a estrutura de comunidades para realizar a detecção de períodos em séries temporais. Este método é robusto para dados ruidosos e não requer parâmetros. Com os métodos e resultados apresentados nesta tese, concluímos que a teoria da redes complexas é benéfica para a mineração de dados em séries temporais. Além disso, esta abordagem pode proporcionar melhores resultados do que os métodos tradicionais e é uma nova forma de extrair informações de séries temporais que pode ser facilmente estendida para outras tarefas.

Palavras-chave: Mineração de séries temporais, Redes complexas, Mineração de grafos, Detecção de comunidades, Agrupamento.

ABSTRACT

FERREIRA, L. N. **Time series data mining using complex networks**. 2017. 147 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, 2017.

A time series is a time-ordered dataset. Due to its ubiquity, time series analysis is interesting for many scientific fields. Time series data mining is a research area that is intended to extract information from these time-related data. To achieve it, different models are used to describe series and search for patterns. One approach for modeling temporal data is by using complex networks. In this case, temporal data are mapped to a topological space that allows data exploration using network techniques. In this thesis, we present solutions for time series data mining tasks using complex networks. The primary goal was to evaluate the benefits of using network theory to extract information from temporal data. We focused on three mining tasks. (1) In the clustering task, we represented every time series by a vertex and we connected vertices that represent similar time series. We used community detection algorithms to cluster similar series. Results show that this approach presents better results than traditional clustering results. (2) In the classification task, we mapped every labeled time series in a database to a visibility graph. We performed classification by transforming an unlabeled time series to a visibility graph and comparing it to the labeled graphs using a distance function. The new label is the most frequent label in the k -nearest graphs. (3) In the periodicity detection task, we first transform a time series into a visibility graph. Local maxima in a time series are usually mapped to highly connected vertices that link two communities. We used the community structure to propose a periodicity detection algorithm in time series. This method is robust to noisy data and does not require parameters. With the methods and results presented in this thesis, we conclude that network science is beneficial to time series data mining. Moreover, this approach can provide better results than traditional methods. It is a new form of extracting information from time series and can be easily extended to other tasks.

Keywords: Time series data mining, Complex networks, Graph mining, Community detection, Clustering.

LIST OF FIGURES

Figure 1	– Partial map of the world wide web extracted in June 1999.	29
Figure 2	– Difference in comparison alignment used by (a) Euclidean distance and (b) <i>Dynamic Time Warping</i> (DTW).	38
Figure 3	– An example of time series clustering. (a) We represented the set of original data in the form of Cartesian points. Each color represents a possible partition. For example, it is possible to find: (b) 2 clusters or (c) 6 clusters. Even different number of partitions can be found.	39
Figure 4	– (a) An example of a dendrogram that represents the result of a hierarchical clustering algorithm of a data set consisting of five points $\{p_1, \dots, p_5\}$. (b) Displays the same dendrogram information using annotation sets. At each iteration of the algorithm, if the clustering is divisive, then both the dendrogram and the order of sets are seen from top to bottom. If the algorithm is agglomerative, then the inverse order is taken from the bottom up.	40
Figure 5	– An example of time series segmentation. From a time series (a) composed by many values and noise, the objective of the segmentation is to find the most important points of the series and to remove the less important ones. (b) Illustrates the main points of the original series. A low-dimensional approximation that preserves the main features of the original series will be obtained by drawing segments between the main points.	51
Figure 6	– An example of a periodic series similar to those obtained by electrocardiograms. In this series, it is possible to notice a subsequence that differs from the pattern (anomaly). Anomaly detection consists of detecting points, subsequences or even complete series that differ from the pattern.	53
Figure 7	– The outlier detection process in temporal data can be subdivided according to the data type: temporal data, data flows, distributed data, spatiotemporal data, and networks.	53
Figure 8	– Motif detection consists of finding subsequences in time series that occur frequently. These subsequences are called motifs. This figure illustrates two examples of motifs in different colors. Motifs can have different sizes and may be overlapping.	57
Figure 9	– An example of time series forecasting. (a) Input time series. (b) The purpose of the forecast is to predict the next h values (red line) of a series. (c) The higher the h , the harder the forecast.	59

Figure 10 – The process of transforming a regular network into a random network without increasing vertices or edges. The initial regular network (left) has 20 vertices, each of them connects to the four closest vertices. The network generation process was repeated twice for different values of p . The small-world network (center) was generated for small values of p . When $p \rightarrow 1$, the result is a random network (right).	66
Figure 11 – An example of a network with three communities represented by colors. Inside each community, the vertices are highly connected. On the other hand, there are few connections between vertices of different communities.	69
Figure 12 – Time series clustering using community detection in networks. First, we construct a network where every vertex represents a time series connected to its most similar time series using a distance function. Then we apply community detection algorithms to cluster time series. The communities are represented by vertices with different colors.	77
Figure 13 – The influence of the parameters (a) k and (b) ε on the resulting number of communities. Weak (gray) lines represent the normalized real variation of the parameter for each combination of data set, time series measure and community detection algorithm. The strongest line (blue) is an interpolation of all results, showing the average behavior. The k -NN construction method just allows discrete values of k while the ε -NN method accepts continuous values. This difference explains why the k -NN interpolation presents the sharpest decrease. In small data sets, k can assume just a few values. However, small variations of k can result in a densely connected network.	81
Figure 14 – An example of the influence of the network construction method on the clustering result for the coffee data set (28 time series divided in 2 classes). In this case, we constructed three k -NN networks with the INTPER measure. Vertices colors represent the communities found with the fast greedy algorithm. (a) $k = 1$ results in a disconnected network where every component is a community (RI=0.64). (b) $k = 7$ creates a connected network with 2 communities that correctly clustered all the time series (RI=1). (c) $k = 27$ creates a fully connected network where the whole network form just one big community (RI=0.48).	82
Figure 15 – Box plot with the best rand distribution divided by measures and networks construction method. The letters in the top denote significant differences. Results that share at least one letter mean that they do not present significant differences using the Nemenyi test and a significance level of .05.	83

Figure 16 – Box plot with the best rand distribution divided by community detection algorithms and networks construction method. The letters in the top denote significant differences. Results that share at least one letter mean that they do not present significant differences using the Nemenyi test and a significance level of .05.	84
Figure 17 – Box plot with the comparison of different time series clustering algorithms applied to different data sets.	88
Figure 18 – The Cylinder-Bell-Funnel (CBF) data set is composed of three groups of series: (a) the cylinder group of series is characterized by a plateau, (b) the bell group by an increasing linear ramp followed by a sharp decrease and (c) the funnel group by a sharp increase followed by a decreasing ramp.	89
Figure 19 – Network representation of the CBF data set using the ϵ -NN construction method ($\epsilon = 58.87$) with DTW. Vertices colors indicate the 3 communities that represent each group of time series illustrated in Fig. 18. All the 30 time series were correctly clustered, except one ($RI = 0.96$). The time series with label “3” belongs to community of color blue (bottom).	89
Figure 20 – The two patterns data set is composed of the sequence of two patterns: an upward (U) step, which goes from -5 to 5 , and a downward (D) step, which goes from 5 to -5 . The order which these patterns occur define each group: (a) UU, (b) UD, (c) DU and (d) DD.	90
Figure 21 – Network representation of the two patterns data set using the ϵ -NN construction method ($\epsilon = 44.91$) with DTW. Vertices colors indicate the four communities that represent each group of time series illustrated in Fig. 20. All the 1000 time series were correctly clustered ($RI = 1$).	91
Figure 22 – Examples of visibility graphs for the series $X = 8, 5, 5, 2, 2, 8, 1, 6, 8, 7, 1, 4, 9, 8, 4, 10, 9, 1, 10, 4$. (a) Natural Visibility Graph (NVG) (b) Horizontal Visibility Graph (HVG).	95
Figure 23 – Time series classification using networks. The idea is to transform labeled training set of time series into a set of networks (different colors represent different labels). Then, an unlabeled time series is transformed into a network and calculated it’s similarity to the other networks from the training data set. Finally, the new time series can be classified using the k -NN algorithm.	99
Figure 24 – A case where 1-NN with DTW fails to classify time series. The goal is to classify the times series 5 and 6 according to its shape similarity to class blue: sin (series 1) or red (2). The 1-NN with DTW algorithm misclassifies it as belonging to class red (2).	100

Figure 25 – Two NVGs constructed using the 2 time series in test dataset in Fig. 24 (numbers 5 and 6 respectively). (a) NGV for all sinusoidal time series in red (1, 3 and 5), (b) NGV for all cos-sinusoidal time series in blue (2, 4 and 6). Using the 1-NN we can correctly classify this time series.	100
Figure 26 – Time series dataset (a) composed by distances between the outline and the image center of beetles (b) and (c) flies.	101
Figure 27 – A case where 1-NN and DTW fail to classify a time series from the BeetleFly dataset. The objective is to classify the time series in red (test 12). Networks and time series with the same color indicate the correspondent transformation. On the left of each network, there is a square that represents the normalized adjacency matrix of the network. The darker the color of the edge, the stronger the connection. DTW considers the time series in green (train 7) as the most similar. Using GHD, the distances between the test instance (red) and the two training instances 7 and 11 are 33.60 and 26.22 respectively. In this case, GHD correctly assigns the label according to similarity.	103
Figure 28 – Result for the transformation process of two different time series into networks. In both cases, the result of community detection (identified by colors) is the same. This feature is used to handle missing data in the series.	109
Figure 29 – Periodicity detection in a sinusoidal dataset. (a) The dataset was generated using the function $\sin(x)$ from 0 to 6π sampled at intervals of 0.10. and divided in $n = 14$ ranges. (b) Network resulted of the transformation from the sinusoidal dataset. Different colors represent the 4 different communities found with the fast greedy algorithm (CLAUSET; NEWMAN; MOORE, 2004). (c) Result of the periodicity detection algorithm. Vertical red lines represent the breaks in the periods found by the method. Every period $P = \{64, 127\}$ in the sinusoidal series was successfully found with the algorithm.	109
Figure 30 – Application of the proposed method to detect periodicity in a real dataset composed by the average monthly temperatures of the city of São Carlos - SP, Brazil. Figure (a) shows the plot of the series divided in $n = 19$ intervals separated with horizontal dashed lines. These intervals are labeled from 1 to 19 where interval 1 is the most low one. Vertices colors represent each community in the networks. (b) Exhibited the network generated from the temperature time series. Colors represent different communities. (c) Shows the result of periodicity detection where every period is separated with red vertical lines.	112

LIST OF TABLES

Table 1 – The UCR time series data sets used in the experiments	79
Table 2 – Time series distance functions used in the experiments	80
Table 3 – Community detection algorithms used in the experiments	81
Table 4 – Performance of the two network construction methods.	82
Table 5 – Performance of different combinations of networks construction methods, distance functions and community detection algorithms.	85
Table 6 – Best time series measures for each of the rival methods	85
Table 7 – Best rand index for each clustering algorithm I	86
Table 8 – Best rand index for each clustering algorithm II	87

LIST OF ABBREVIATIONS AND ACRONYMS

ε -NN	ε -nearest neighbors
k -NN	k -nearest neighbors
ANN	Artificial neural networks
ARIMA	Autoregressive integrated moving average
CID	Complexity-Invariant distance
COR	Pearson Correlation distance
DTW	Dynamic time warping distance
DWT	Wavelet Transform distance
ED	Euclidean distance
ERP	editing distance with real penalty
GHD	Gromov-Hausdorff distance
HMM	Hidden Markov models
IBL	Instance-based algorithms
INTPER	Integrated Periodogram distance
LCSS	longest common subsequence
NVG	Natural visibility graph
PLA	Piecewise linear approximation
RI	Rand index
RNN	Recurrent neural networks
SOM	Self-organizing maps
STS	Short Time Series distance
SVM	Support vector machine
VG	Visibility graph

LIST OF SYMBOLS

X — Time series

S^X — Subsequence of the time series X

W^X — Set with all the subsequences of time series X extracted by a sliding window process

$d(Q, C)$ — Distance function between two objects Q and C

D — Distance matrix

G — Graph or network

(i, j) — Edge that connects the vertices i and j

w_{ij} — Edge weight between the vertices i and j

A — Adjacency matrix

m — Number of edges in a graph

n — Number of vertices in a graph

k_i — Degree of vertex i

N_i — Neighbors of vertex i

$\langle k \rangle$ — Average degree

$p(k)$ — Degree distribution

l — Mean Shortest paths

c — Clustering coefficient

r — Assortativity

Q — Modularity

B_u — Betweenness centrality of vertex u

C — Partitions of a dataset

Y — Set of labels

CONTENTS

1	INTRODUCTION	27
1.1	Objectives	30
1.2	Motivation	31
1.3	Thesis contributions	33
1.4	Thesis outline	34
2	TIME SERIES DATA MINING	35
2.1	Introduction	35
2.2	Preprocessing	37
2.3	Distance function	37
2.4	Time series data mining tasks	39
2.4.1	<i>Time series clustering</i>	39
2.4.1.1	<i>Complete time series clustering</i>	43
2.4.1.2	<i>Subsequence clustering</i>	44
2.4.2	<i>Time series classification</i>	46
2.4.3	<i>Time series segmentation</i>	50
2.4.4	<i>Anomaly detection</i>	52
2.4.4.1	<i>Outliers detection in time series</i>	53
2.4.4.2	<i>Outliers detection in temporal networks</i>	55
2.4.5	<i>Motifs detection</i>	57
2.4.6	<i>Forecasting</i>	58
2.5	Final remarks	61
3	COMPLEX NETWORKS	63
3.1	Introduction	63
3.2	Complex networks models	65
3.2.1	<i>Random networks</i>	65
3.2.2	<i>Small-world networks</i>	66
3.2.3	<i>Scale-free networks</i>	67
3.3	Complex network measures	67
3.4	Community detection	69
3.5	From time series to complex networks	72
3.6	Final remarks	74

4	TIME SERIES CLUSTERING USING NETWORKS	75
4.1	Introduction	75
4.2	Clustering time series with networks	77
4.3	Experimental evaluation	78
4.3.1	<i>Experiment settings</i>	78
4.3.2	<i>Network construction influence</i>	81
4.3.3	<i>Time series distance function influence</i>	82
4.3.4	<i>Community detection algorithm influence</i>	83
4.3.5	<i>Comparison to rival Methods</i>	84
4.3.6	<i>Detecting time series clusters with time-shifts</i>	88
4.3.7	<i>Efficiency to detect shape patterns</i>	88
4.4	Final remarks	91
5	TIME SERIES CLASSIFICATION USING NETWORKS	93
5.1	Background and related works	94
5.1.1	<i>Visibility graphs</i>	94
5.1.2	<i>Graph similarity</i>	96
5.1.3	<i>Graph classification</i>	97
5.1.4	<i>Time series classification</i>	97
5.2	Method description	98
5.3	Experimental evaluation	99
5.3.1	<i>Toy case</i>	99
5.3.2	<i>Real case</i>	101
5.4	Final remarks	102
6	PERIODICITY DETECTION USING NETWORKS	105
6.1	Introduction	105
6.2	Related works	106
6.3	Method description	107
6.4	Experimental evaluation	110
6.4.1	<i>Sinusoidal series</i>	110
6.4.2	<i>Temperature series</i>	110
6.5	Final remarks	111
7	CONCLUSION	113
7.1	Future works	114
7.2	Publications	115
	BIBLIOGRAPHY	117

INTRODUCTION

A time series is a set of ordered values in time (MORETTIN; TOLOI, 2006). Examples of time series are the daily temperature, the stock price variation, heart beats per second, etc. It is possible to observe these sets of ordered data in many fields of science. Therefore, the study of time series has several applications in many different areas of knowledge, such as robotics (RAMONI; SEBASTIANI; COHEN, 2000), space science (VLACHOS *et al.*, 2003), speech recognition (TRAN; WAGNER, 2002), energy consumption (CUARESMA *et al.*, 2004), medicine (BURKOM; MURPHY; SHMUELI, 2007), genetics (LIN; KAMINSKI; BAR-JOSEPH, 2008), ecology (LI *et al.*, 2001), music (WEIHS; LIGGES; SOMMER, 2006), finances (SONG; LI, 2008), criminology (GORR; OLLIGSCHLAEGGER; THOMPSON, 2003), hydrology (OUYANG *et al.*, 2010), seismology (MORALES-ESTEBAN *et al.*, 2010), meteorology (ROMILLY, 2005), and many others.

The systematic study of time-correlated data is called time-series analysis (SHUMWAY; STOFFER, 2011). It uses two different approaches of analysis: time and frequency. In the temporal approach, it is considered that an observed data depends on another data that was previously observed. In this type of approach, the idea is to construct mathematical models that describe the series as a function of time and that can be used to make predictions. The frequency approach performs a transformation of the series into the frequency spectrum and uses models to decompose the series into simpler parts to find periodicity in the observations of the series. This strategy can be used to find cycles and describe seasonality. Among the most common forecasting methods, we cite the Autoregressive Integrated Moving Average (ARIMA) (BOX; REINSEL; JENKINS, 1994), decision trees (BREIMAN *et al.*, 1984), k -nearest neighbors (ALTMAN, 1992), artificial neural networks (ANN) (BISHOP, 1995), bayesian networks (NEAPOLITAN, 2003), support vector machines (SVM) (DRUCKER *et al.*, 1996), Hidden Markov Models (HMM) (CAPPÉ; MOULINES; RYDEN, 2005), and Kalman filters (HARVEY, 1990). The most used techniques to study the frequency domain are the Fourier transform (BLOOMFIELD, 2004) and wavelets (PERCIVAL; WALDEN, 2000).

A new approach to the study of sequential data is the time series data mining. It can be defined as the extraction of non-trivial information from a database (LAXMAN; SASTRY, 2006). Unlike time series analysis, data mining techniques traditionally treat temporal data as an unordered collection of events and ignore temporal information (MITSA, 2010). The study scope is also not limited to series forecasting and control. In temporal data mining, the study is divided into seven tasks (ESLING; AGON, 2012): (1) *Search*: given a search query, this task consists of searching a database for a time series that satisfies the query (FALOUTSOS; RANGANATHAN; MANOLOPOULOS, 1994); (2) *Motif detection*: consists of searching recurrent subsequence (motifs) in a time series (LIN *et al.*, 2004). Periodicity detection is also related to this task; (3) *Anomaly detection*: this task consists of finding subsequences in the series that differ from the standard pattern (WEISS, 2004); (4) *Segmentation*: Consists of finding approximations for the series in order to reduce its dimensionality without losing its fundamental characteristics (KEOGH *et al.*, 1993); (5) *Clustering*: this task can be divided into two subtasks, the first approach is to group a set of time series of a database and the second one consists on grouping subsequences of one or more series (FU, 2011); (6) *Classification*: This task consists of using a set of previously classified (labeled) series and training an algorithm to assign labels to an unlabeled series (BAKSHI; STEPHANOPOULOS, 1994); (7) *Forecasting*: Consists of using a model to capture temporal correlation in order to predict future values (WEIGEND; GERSHENFELD, 1994).

Different models have been used to study time series. A new way of modeling and analyzing time series is through complex networks. With this approach, it is possible to describe the interaction between the vertices of the network and the dynamics that govern them (STROGATZ, 2001). This approach has been used in several fields of science due to the capability of modeling different complex systems in a unified form (NEWMAN, 2003; SAYAMA *et al.*, 2013). Therefore, it is possible to study different types of networks and find common properties. Complex networks are usually formed by a large number of connected vertices forming a non-trivial topology (BOCCALETTI *et al.*, 2006). Examples of complex networks are the social networks (WASSERMAN; FAUST; IACOBUCCI, 1994; SCOTT, 2000), collaboration network in scientific articles (BARABÁSI *et al.*, 2002), World Wide Web (WWW) (HUBERMAN, 2003), the wired internet (Fig. 1) (FALOUTSOS; FALOUTSOS; FALOUTSOS, 1999), email exchanges network (EBEL; MIELSCH; BORNHOLDT, 2002; NEWMAN; FORREST; BALTHROP, 2002), phone calls (AIELLO; CHUNG; LU, 2000; AIELLO; CHUNG; LU, 2002), power distribution networks (WATTS; STROGATZ, 1998), airlines (AMARAL *et al.*, 2000), neural networks (SPORNS, 2002), and interaction between proteins (MASLOV; SNEPPEN, 2002). Furthermore, other systems can be modeled as networks, such as dynamic systems and time series (DONNER *et al.*, 2011). For example, vertices can represent the subsequences of the series and the edges represent the correlation between two different subsequences (YANG; YANG, 2008).

After representing the domain by a network, it is possible to study the network topology

Figure 1 – Partial map of the world wide web extracted in June 1999.

Source: Cheswick, Burch and Branigan (2000).

and dynamics to extract relevant information. This study originated in the late 1950s with the work of Erdős and Rényi (1959) that studied random networks and proposed a model for their generation. However, this random model does not explain many real networks. In social networks, new friendships are not formed at random. People tend to meet close people, i.e., people who are friends in common. Another property of social networks is that people taken at random are on average separated from each other by just a few other people. This feature is called “small world” and had motivated the creation of a network model with the same name. This model was proposed by Watts and Strogatz (1998) and has the capability of generating networks that present the same characteristics of social networks. At the same time, Barabasi and Albert (1999) demonstrated that several real networks show a topology formed by many vertices with few connections and few vertices with many connections (hubs). They also observed that the probability distribution of the number of connections by vertices (degree) of these networks follows a power law distribution $p(k) \sim k^{-\lambda}$, where λ is a constant that varies between 2 and 3. In random networks, the degree distribution follows a Poisson and the average degree $\langle k \rangle$ indicates the scale of the network, since a randomly chosen vertex will have degree $k = \langle k \rangle \pm \langle k \rangle^{\frac{1}{2}}$. On the other hand, in the networks studied by Barabasi and Albert (1999), the average degree cannot be used as a scale to predict the degree of a vertex taken at random. That is why this model is called

free-scale networks.

Among the many different properties of real networks, one of the most studied is the structure of communities. These structures are groups of densely connected vertices (communities) with a few connections between this groups and others. Communities appear in several types of real networks (PORTER; ONNELA; MUCHA, 2009). In networks of protein interaction, communities can be used to classify unknown proteins (GIRVAN; NEWMAN, 2002). Other examples of real networks that have the same property are the social networks (TRAUD *et al.*, 2011), email exchange (TYLER; WILKINSON; HUBERMAN, 2003), food chains (GIRVAN; NEWMAN, 2002), and correlation networks between time series of the stock market (BONANNO; VANDEWALLE; MANTEGNA, 2000). The community detection problem is analogous to the graph partition problem in graph theory. The latter one is an NP-Complete problem (FORTUNATO, 2010). However, many heuristics have been developed to find approximately optimal solutions. These algorithms use different approaches to detect communities. Examples include the modularity maximization (CLAUSET; NEWMAN; MOORE, 2004), spectral analysis (DONETTI; MUÑOZ, 2004), random walks (PONS; LATAPY, 2005), label propagations (RAGHAVAN; ALBERT; KUMARA, 2007), information compression (ROSVALL; BERGSTROM, 2008), particles competition (QUILES *et al.*, 2008), collective intelligence (OLIVEIRA; ZHAO, 2008), Oscillator synchronization (ZHAO; CUPERTINO; JR., 2008), and consensus (CUPERTINO; HUERTAS; ZHAO, 2013).

1.1 Objectives

The objective of this thesis is to study and propose solutions for time series data mining using complex networks. In this approach, we map time series to the topological space and explore them using network techniques. The central question behind this research is: “*Can complex networks bring advantages to time series data mining?*”. To tackle this general question, we have broken it into the tasks of time series data mining. In this thesis, we decided to treat three tasks: clustering, classification and periodicity detection. In the following, we present the specific objectives:

- *Time series clustering*: In this task, the objective is to propose a time series clustering technique based on community detection in networks. The hypothesis is that community detection algorithms could better detect groups of similar time series. To check this hypothesis, we verified what are the best combination of network construction method, time series distance function, and community detection algorithm. Then, we used these methods to propose new time series clustering method. We wanted to check if this new approach would present better results than traditional methods.
- *Time series classification*: The goal here is to propose a time series classification method based on graphs. The idea is to transform a set of time series to the topological space. This

transformation process permits the use of the whole graph theory and network science to help in the classification process. Previous works (WANG *et al.*, 2013) show that the k -NN classifier is very difficult to beat. This classifier uses a distance function to compare the unlabeled time series to the labeled ones. What we wanted is to transform the time series comparison problem into a graph matching problem. The hypothesis was that the difference between classes would be translated to the topology of the resulting networks.

- *Periodicity detection*: The goal was to propose a method for periodicity detection in time series using network science. Here again, we decided to transform a time series to the topological space to use network techniques to find periodicity. There are different approaches to transforming a time series into a graph. Some of them generate interesting graph features. For this task, we take advantage of a transformation method and use it to propose a new method to detect periodicity.

1.2 Motivation

The study of time series is interesting for many fields of science. In many of these areas, there is an interest in extracting and analyzing large amounts of sequential data. The general objective is to be able to control specific applications in different areas of knowledge. Thus, the development and improvement of techniques that solve data mining tasks are fundamental. The main tasks are the detection of anomalies, detection of periodicities, trend detection, prediction, grouping, classification, signal processing and control of dynamic systems. Understanding how time series are generated and how their dynamics occur can enable the extraction of knowledge that can be used for decision making.

From the characterization works of complex networks (COSTA *et al.*, 2007), it was possible to study several real systems based on networks. These studies allowed a better understanding of the relationship between network structure and dynamics (BOCCALETTI *et al.*, 2006; NEWMAN; BARABASI; WATTS, 2006). The successful application of network theory in many fields of science motivated the use of this theory to study time series (ZHANG; SMALL, 2006; YANG; YANG, 2008; LACASA *et al.*, 2008; Donner *et al.*, 2009; MARWAN *et al.*, 2009; CAMPANHARO *et al.*, 2011; SMALL, 2013; ZOU *et al.*, 2014). Complex networks can capture important complementary characteristics that the traditional methods of time series analysis are not able to detect (DONNER *et al.*, 2010). Moreover, this new model might find new patterns that can be used to develop new techniques for data mining in time series.

Information extraction from connected structures, such as networks and graphs, is by itself a challenge (GETOOR, 2003). One of the primary motivations for studying structured data in networks is that connections (edges) between data (vertices) can demonstrate certain patterns that may be useful for many data mining tasks, but which are hardly captured by traditional statistical models. Once the data are represented as a network, new specific tasks arise in the

data mining process. The main tasks are the identification of edges types, prediction of the force (weight) of the connections and the estimation of the number of smaller paths between vertex pairs (GETOOR, 2003). In addition to these, classification and clustering tasks should also take into account the links between the data. In summary, the network structure presents additional information in the relationship (edges) between the data. Using this information is the biggest challenge in this type of data mining.

One challenge in time series data mining is the processing of large data series (YANG; WU, 2006). As much as the technological advance has brought faster processors, the amount of data generated has increased exorbitantly. This huge amount of data makes impossible to use many data mining techniques due to processing time. Complex networks focus on studying real networks, which in most cases are large and dynamic. Thus, most of the methods developed for extracting information from networks take into account these two factors. Therefore, these techniques are suitable for time series data mining and might be a useful tool to evaluate large series.

Another problem with conventional data mining techniques involves noise (YANG; WU, 2006). Noises are imperfections or poor data quality, including missing data, inconsistent values, duplicate data, or measurement errors. Considering that it is not possible to avoid noise in the data, there are two ways to deal with such problems. The first approach is to preprocess the data to correct the problems. The second option is to build robust data mining techniques that tolerate data noise (TAN; STEINBACH; KUMAR, 2005). Many of the existing temporal data mining techniques are not able to extract information from noise data, which reduces the effectiveness of prediction algorithms. The use of complex networks for the study of time series allows obtaining more information about the series than the traditional techniques. Consequently, it might permit the development of techniques that are more effective against the noise present in the data.

Considering that time series are composed of large amounts of data, many grouping techniques preprocess the data to extract feature vectors. However, during this process, some information about the series may be lost. This same problem occurs when traditional data mining methods are applied to structured databases, such as graphs, trees, or relational tables (COOK; HOLDER, 2006). It may occur because conventional algorithms do not take into account the relationships in the data. These techniques require that the structured data should be first transformed into an unstructured data. During this process, the information contained in the relationships is lost (DŽEROSKI, 2003). In complex networks, several clustering algorithms (community detection) have been proposed to find groups of vertices without having to transform them, i.e., directly from the network structure (FORTUNATO, 2010). Furthermore, most of them are designed to process large amounts of data. Thus, applying these algorithms to the task of grouping time series can bring better results.

In the time series classification task, the best results were obtained using the k -NN ($k = 1$) algorithm with the dynamic time warping (DTW) distance (XI *et al.*, 2006). One natural question

here is: is this the best possible result? The DTW distance first aligns two series and compare them. This alignment problem is analogous to the graph matching problem in graph theory, a well-known problem. The similarity between the two problems motivated us to transform time series into networks and take advantage of graph theory to classify time series.

Another interesting problem is the search for recurrent patterns in time series. Many clustering techniques were used in the attempt to group similar subsequences (Sec. 2.4.1.2). However, a study by Keogh, Lin and Truppel (2003) has shown that using this approach to search for patterns in series yields insignificant results. Regardless of the time series, the results will be similar. It happens because the used subsequences were generated from sliding windows. This deficiency motivates the use of alternative techniques for the modeling of subsequences. A new way of representing temporal data is through networks. This mapping of series to a topological space can preserve essential characteristics of the series that other models cannot. Also, once represented in the form of a network, it is not necessary to adapt clustering algorithms to work with networks, since many community detection algorithms have already been proposed (Sec. 3.4) and can be used to group similar vertices (subsequences).

1.3 Thesis contributions

Here we present how we tackled the objectives proposed in this thesis.

- *Time series clustering*: We proposed a network-based method for whole time series clustering. The idea was to map every time series from a database to a network. We represent each time series by a vertex and we connect two vertices if the two underlying time series are similar. The similarity was measured using any time series distance functions. After the network construction, we used community detection algorithms to find groups of highly connected vertices. We tested different combinations of distance functions, network construction methods, and community detection algorithms to verify which combination provides the best result. We compared the results to traditional clustering algorithms and observed that this approach is statistically better than traditional clustering methods. In chapter 4 we presented these results in details.
- *Time series classification*: We propose a new time series classification method based on graphs. We chose the visibility graph algorithm to transform a time series into a graph. One main feature of this transformation method is that affine transformations from one time series lead to the same graph. Another point is that small variations (noise) in the time series may result in the same graph. These features are interesting when time series from the same class are affine transformations of one another. In literature, the k -NN classifier with DTW is considered very difficult to beat. This time series distance function first aligns two series before comparing them. In our method, we used an analogous approach. We

transformed the time series alignment problem into a graph matching problem. We also used the k -NN classifier but with the Gromov-Hausdorff distance (GHD) for graphs. In chapter 5, we show that this approach can beat DTW in specific situations.

- *Periodicity detection*: We propose a new network-based method for periodicity detection in time series. We also here take advantages of the visibility graph transformation. This process maps local maxima to highly connected vertices that connect two different communities. We proposed a method that locates these vertices using community detection and breaks the network into communities. The vertices from each community represent the values of a period in the series. This method does not require parameters and is robust to noise. In chapter 6, we present details about this method and the application in a real problem.

1.4 Thesis outline

In the following, we present how this text is organized. This thesis has 7 chapters. In the Introduction (Chap. 1), we contextualized this thesis. We also briefly presented some related topics, the objectives, the motivation and the contributions of this thesis. The next two chapters, 2 and 3, we present the two main background topics to this thesis: time series data mining and complex networks, respectively. In these two chapters, we introduce some topics that are necessary for understanding the later chapters. We also show the literature review about the two main research topics related to this thesis. In Chapter 4, we show the results for time series clustering. We present the problem, the related works and how our clustering methods works. We also show some experimental results. In Chapter 5, we present how to use graph theory and graph mining to make time series classification. We present our new method and the experimental results. In Chapter 6, we show our network-based method for time series periodicity detection and the application of this method in a real problem. In Chapter 7, we present the conclusions of the results obtained during the elaboration of work here presented. Finally, the bibliographic references used for the development of this thesis are presented.

TIME SERIES DATA MINING

2.1 Introduction

Data mining is the process of automatically extraction of knowledge from large databases using machine learning, pattern recognition, database techniques, and statistics (ZAKI; JR, 2014). It is part of a broader process of knowledge extraction that can be divided into 3 stages: (1) Preprocessing of data to make them appropriate; (2) Application of pattern extraction algorithms (data mining); (3) Post-processing, which consists of interpreting the results, validating and confirming hypotheses. Data mining involves tasks as sorting, clustering, merging, summarizing, anomaly detection, and discovering frequent patterns.

A data set is formed by a collection of objects, which can also be called registers, vectors, cases, patterns or examples. Each object in the data set is represented by a set of characteristics, also called attributes or variables. These characteristics vary from object to object and define each set of data. The attributes that describe the data can be of different types: binary, categorical, numeral, interval, ordinal, relational, textual, spatial, temporal, spatiotemporal, images, multimedia or mixtures (LIAO, 2005).

Definition 1 (Attribute). An attribute is a property or characteristic of an object that can vary, either from one object to another or from time to time.

In addition to the attributes that define the data, another important characteristic is the form they are stored. The data may be stored in a text file or a database table. Or, it can be stored in more complex structures such as trees, graphs and XML documents. The data structure is important because it defines the technique that will be used for data analysis. Most of the data mining techniques have been developed for unstructured data sets. Therefore, it is usually necessary to transform the data into unstructured forms. The main problem with this process is that it loses the relationship between the data. Thus, there is a growing interest in methods of information extraction that do not require this transformation (DŽEROSKI, 2003). In this thesis,

we will focus on extracting information from data stored in two specific structures: time series and networks.

A time series is a set of sequential values in time. If the time interval is fixed, this series is called discrete. On the other hand, if for every instant $t \in \mathbb{R}_+$ there is an observation, then the series is said to be continuous. In many real cases, variables are measured (sampled) over time resulting in discrete series. Some examples are the heart rate per second, monthly rainfall levels, daily stock price, etc. Also, a continuous time series can also be transformed into a discrete series by applying some discretization technique. Each observation of the series can be composed of one or more variables and can be classified as univariate and multivariate, respectively. For the sake of brevity, we use the single term “series” in this text in reference to a time series.

Definition 2 (Discrete Time Series). A time series X is an ordered sequence of n real variables:

$$X = \{x_1, x_2, x_3, \dots, x_n\}, x_i \in \mathbb{R}, i \in \mathbb{N}.$$

Definition 3 (Subsequence). Given a time series $X = \{x_1, \dots, x_n\}$ of length n , a subsequence $S^X = \{x_k, x_{k+1}, \dots, x_{k+m-1}\}$ is a time series of length $m \leq n$ composed of contiguous elements of X where $1 \leq k \leq n - m + 1$.

Definition 4 (Sliding window). Given a time series X of size n and an arbitrary parameter w that defines the size of the time window, it is possible to construct an array W^X with all subsequences extracted from X using a window slider. The matrix W^X can be written as follows:

$$W^X = \begin{bmatrix} x_1 & x_2 & \cdots & x_w \\ x_2 & x_3 & \cdots & x_{w+1} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n-w+1} & x_{n-w+2} & \cdots & x_n \end{bmatrix},$$

where the j -nth line is $W_j^X = \{x_j, x_{j+1}, \dots, x_{j+w-1}\}$ such that $1 \leq j \leq n - w + 1$ and $j \in \mathbb{N}$. The size of matrix W^X is $n - w + 1$ rows and w columns.

Definition 5 (Time series database). A time series database DB is a set of time series.

Definition 6 (Time series model). Given a time series $X = \{x_1, \dots, x_n\}$ of size n , a model $\bar{X}$ is a representation of X with reduced dimensionality $\bar{d}$ such that $\bar{d} \ll n$ and $\bar{X}$ approximate X .

The main components of a time series are the trend, seasonality, cycles, and noise (random influence) (MORETTIN; TOLOI, 2006). The trend is the gradual increase or decrease that occurs in the values of the observations. Seasonality represents the fluctuations of the series over a long period. The cycles are similar to seasonality but have a longer duration. Generally, in time series measured in days or weeks, the seasonality is annual. Thus, the length of the component period is directly related to the length of the series. If it is possible to precisely

predict one value in the series using past observations, then the series is considered deterministic. However, in most real series, future values are only partially defined by past observations. The random component is responsible for the stochasticity of most of the real series. These series are called stochastic or random.

2.2 Preprocessing

Very often, the data set has imperfections (noise) generated due to measurement, storage or processing errors (PYLE, 1999). These errors end up introducing incorrect data that might compromise the performance of data mining algorithms. Another common error is the absence of some values in the data set. The primary purpose of the preprocessing step is to solve these problems. In time series, the noise can be handled with traditional signal processing techniques such as digital filters and wavelets (GAO *et al.*, 2010a).

Another objective of this step is to transform the data to enable or make more efficient the execution of the data mining algorithms. In time series of different sizes, re-sampling is a preprocessing technique that matches the dimensions of the series. Another transformation technique is normalization, which can be used to match the scales of different series (CHAKRABARTI *et al.*, 2002).

2.3 Distance function

A common concept used in several data mining techniques is similarity. It is defined as a function $d : Q \times C \rightarrow [0, \infty)$, which determines how different two objects Q and C are. Values close to 0 mean that the two objects are similar. The terms “similarity” and “distance” will be used in this thesis to express opposite ideas.

Definition 7 (Distance measure). The distance measure $d(Q, C)$ is a function that has as input two objects Q and C and returns the distance between them. In addition, this function should present the following properties:

- *Symmetry*: $d(Q, C) = d(C, Q), \forall C, Q$.
- *Positivity*: $d(Q, C) \geq 0, \forall C, Q$.
- *Triangular difference*: $d(Q, C) \leq d(Q, K) + d(K, C), \forall Q, C, K$.
- *Reflection*: $d(Q, C) = 0 \iff Q = C$.

Definition 8 (Corresponding subsequences). Given two subsequences S_1 and S_2 and a distance measure $d(S_1, S_2)$, S_1 and S_2 are said corresponding if $d(S_1, S_2) \leq \epsilon_s$, where ϵ_s is an arbitrary limiting error. When $d(S_1, S_2) = 0$, then S_1 and S_2 are called exact motifs. When $0 < d(S_1, S_2) \leq \epsilon_s$, then S_1 and S_2 are called approximate motifs.

In the context of temporal data mining, several distance functions were created or adapted (DING *et al.*, 2008; ESLING; AGON, 2012). Among them, the Euclidean distance (FALOUTSOS; RANGANATHAN; MANOLOPOULOS, 1994) is one of the most used one. Considering two series X_1 and X_2 , both with equal size n , the Euclidean distance between them can be calculated using the equation 2.1.

$$d(X_1, X_2) = \sqrt{\sum_{i=1}^n (X_1[i] - X_2[i])^2}. \quad (2.1)$$

The biggest problem with the Euclidean distance occurs when there is a distortion in time, as in Fig. 2. The dynamic time warping (DTW) distance was proposed to solve this problem (BERNDT; CLIFFORD, 1994). The DTW does not require that the two series have the same size. The main idea of this measure is to align the two series and get the distance. One issue involving DTW is the fact that programmers generally implement it using dynamic programming and therefore has quadratic complexity, requiring high computational time. Among other distance functions between series are the distance based on the longest common subsequence (LCSS) (VLACHOS; KOLLIOS; GUNOPULOS, 2002), editing distance (Levenshtein) with real penalty (ERP) (CHEN; NG, 2004), editing distance (Levenshtein) in real subsequences (EDR) (CHEN; ÖZSU; ORIA, 2005), DISSIM (FRENTZOS; GRATSIAS; THEODORIDIS, 2007), Sequence weighted alignment (Swale) (MORSE; PATEL, 2007), Spatial Assembling Distance (SpADe) (CHEN *et al.*, 2007), search based on Threshold Queries (TQuEST) (ASSFALG *et al.*, 2006) and Campana-Keogh (CK-1) (CAMPANA; KEOGH, 2010). For a revision and comparison between measures, We point the interested reader to the works of Ding *et al.* (2008) and Serrà and Arcos (2014).

Figure 2 – Difference in comparison alignment used by (a) Euclidean distance and (b) *Dynamic Time Warping* (DTW).

Source: Adapted from Ratanamahatana and Keogh (2004).

2.4 Time series data mining tasks

In this section, we present a review of the main tasks involving time series data mining. Esling and Agon (2012) divide them into seven tasks: search, motif detection, anomaly detection, segmentation, clustering, classification, and forecasting. Each of these will be presented in this section.

2.4.1 Time series clustering

One of the most common tasks of data mining is clustering analysis. The goal is to group the data using only the information in the data and their relationships (TAN; STEINBACH; KUMAR, 2005). The idea is to try to group data that is similar (or related) to each other and different (or unrelated) from other groups. The greater the similarity between the elements of the group and the greater the difference between the groups, the better will be the grouping. However, finding the correct number of groups is the main difficulty of the algorithms. In the figure 3 is illustrated a data set formed by 18 points and two possible results for the process of grouping data. Visually, both solutions may be correct.

Figure 3 – An example of time series clustering. (a) We represented the set of original data in the form of Cartesian points. Each color represents a possible partition. For example, it is possible to find: (b) 2 clusters or (c) 6 clusters. Even different number of partitions can be found.

Source: Elaborated by the author.

Han, Kamber and Pei (2011) classify clustering algorithms into five categories: hierarchical methods, partitioning methods, density-based methods, grid-based methods, and model-based methods. These methods can generate fixed (unique) groups or fuzzy groups (non-unique). Fixed groupings form the vast majority of algorithms that consist of assigning an element to just one group. On the other hand, the fuzzy algorithms can assign an object to more than one group. In the later case, the algorithms usually return the probabilities of one element to belong to one group. The main approaches used by the clustering algorithms are:

- *Hierarchical methods*: The hierarchical algorithms group the data into a hierarchy of groups that can be represented by a binary tree called dendrogram (Fig. 4) (WITTEN; FRANK; HALL, 2011). There are two approaches: agglomerative and divisive methods. (1) In the agglomerative (bottom-up) approach, each object is initially placed in a different group. Then, the algorithms iteratively join the two more similar groups until all elements are in

only one group or if they met some other stopping criteria. (2) In the divisive (top-down) approach, all data is initially considered an entire group that is partitioned until each object is in a different group. Among the main hierarchical methods is the algorithm *single-linkage* (SNEATH, 1957) which, at each iteration, join the groups by observing the shortest distance between data from different groups. Similarly, the *complete-linkage* (SØRENSEN, 1948) and *average-linkage* (SOKAL; MICHENER, 1958) algorithms merge groups by according respectively to the largest distance and the mean of the distances between elements of distinct groups. Among other hierarchical algorithms are the CURE (GUHA; RASTOGI; SHIM, 1998), BIRCH (ZHANG; RAMAKRISHNAN; LIVNY, 1996), and Chameleon (KARYPIS; HAN; KUMAR, 1999). The main advantage of hierarchical methods is that they allow the evaluation of the result for different numbers of groups. For applications where different numbers of groups are required, the hierarchy eliminates the need to rerun the algorithm for a different number of clusters. One disadvantage of this approach is that, if at any stage of the hierarchy construction one group is incorrectly merged (or partitioned), the clustering method will not correct that error in the next iterations, and usually results in following errors. Another disadvantage is the high computational time required to build the hierarchy of groups.

Figure 4 – (a) An example of a dendrogram that represents the result of a hierarchical clustering algorithm of a data set consisting of five points $\{p_1, \dots, p_5\}$. (b) Displays the same dendrogram information using annotation sets. At each iteration of the algorithm, if the clustering is divisive, then both the dendrogram and the order of sets are seen from top to bottom. If the algorithm is agglomerative, then the inverse order is taken from the bottom up.

Source: Elaborated by the author.

- *Partitional methods*: Unlike the hierarchical algorithms, where groups are divided or joined in each iteration, in partitioned methods, the data is partitioned directly into k groups (XU; WUNSCH, 2005). These methods do not provide a clustering hierarchy. Evaluating all possible partitions and choosing the best one among them is usually not an option because

it demands high computational time (LIU, 1968). The two most important heuristics to find partitions in the data are the k -means (FORGY, 1965; MACQUEEN, 1967; BOCK, 2007) and the k -medoids (KAUFMAN; ROUSSEEUV, 1987; KAUFMAN; ROUSSEEUV, 2005). In the former, the mean value of the group represents each cluster. In the latter, the most central objects of the group represent the cluster. These two algorithms also have versions to detect *fuzzy* partitions. They are the fuzzy k -means (BEZDEK, 1981) and *fuzzy c*-medoids (KRISHNAPURAM; JOSHI; YI, 1999). Among other algorithms are the CLARA (KAUFMAN; ROUSSEEUV, 2005), CLARANS (NG; HAN, 2002), and k -modes (HUANG, 1997). The main characteristic of these methods is the efficiency in finding groups of spherical shape in three-dimensional spaces.

- *Methods based on density*: The clustering algorithms based on density consider that a data group is a dense region of objects surrounded by a low-density areas. Thus, the algorithms that use this approach consist in iteratively increasing the size of the group until the density around the group decreases, i.e., until it reaches a user-defined lower limiting value. Examples of algorithms that use this approach are the DBSCAN (ESTER *et al.*, 1996), OPTICS (ANKERST *et al.*, 1999), and DENCLUE (HINNEBURG; KEIM, 1998). The main advantage of these methods is that the user does not need to specify a priori the number of the desired number of groups. These algorithms define this number automatically. Instead of specifying the desired number of groups, the user informs the maximum density of groups. Another advantage is that they can detect clusters in a variety of ways. However, they present problems when the density of the groups differ considerably. Also, these methods are very sensitive to the parameter that limits the density of points. If it is too high, groups are lost. If too little, two groups that should be separated end up in the same group.
- *Methods based on grid*: Grid-based algorithms consist of creating a grid structure and partitioning the data space into a finite number of cells. Then, they calculate the data density in each of the cells and delete cells that have a density less than a user-defined parameter. These methods observe adjacent cells with high data density to form the groups. Among the most commonly used algorithms are STING (WANG; YANG; MUNTZ, 1997), CLICK (AGRAWAL *et al.*, 1998), and MAFIA (BURDICK; CALIMLIM; GEHRKE, 2001). The advantage of this approach is that algorithms can be highly efficient if the input does not create empty cells and if appropriate data structures are used to represent adjacent cells (TAN; STEINBACH; KUMAR, 2005). The disadvantages are similar to those of density based methods.
- *Methods based on models*: They assume that models generate the groups and try to fit the better model into the data set. There are two strategies for these methods based: statistical-based and neural networks based models. Statistical models assume that some probability distributions rule the clustering formation. The best-known example is the EM

algorithm (HARTLEY, 1958; DEMPSTER; LAIRD; RUBIN, 1977), which considers that data follows Gaussian distributions. Another algorithm is the AutoClass (CHEESEMAN; STUTZ, 1996), which uses Bayesian models (CONGDON, 2007). The other approach of the model-based methods are the neural networks (HAYKIN, 2009). Among the most used network models are the ART (CARPENTER; GROSSBERG, 1987) and SOM (KOHONEN, 1990; KOHONEN, 2001) networks.

The data clustering process consists of four steps (XU; WUNSCH, 2008):

1. *Attribute selection:* In the first step, some feature selection method can be applied to extract useful attributes from the data set. The purpose of this selection is to remove unnecessary features and reduce the cost of the clustering algorithms. It also might facilitate the process of grouping data;
2. *Clustering algorithm selection:* In the second step, the user defines the clustering algorithm. This choice depends on the type of data set. Since most algorithms use similarity function, the user also defines this function in this step. Selecting a suitable algorithm and appropriate parameters can result in better partitions. However, there is no universal method that is efficient in any case;
3. *Validation:* Clustering algorithms can always find groups even when there is no real group structure in the data. Different algorithms may result in different groupings. Even the same algorithm can return different clusters if it executes with distinct parameters. Thus, it is necessary to distinguish what is a good and a bad partition. This distinction is usually made by an evaluation measures that verifies the quality of the clustering result and provides a degree of reliability to the user. These measures should only evaluate the clustering result regardless of the algorithm. They are used to answer questions such as how many groups exist in the data or whether the result is really meaningful from a practical point of view;
4. *Interpretation:* After obtaining the clusters, the last step consists of analyzing the result and provide the user with a meaningful knowledge that can develop a clear understanding of the data.

Esling and Agon (2012) subdivide the task of time series data clustering into two subtasks: complete series and subsequences clustering. We briefly present each of these tasks and the main techniques developed for each task in the sections 2.4.1.1 and 2.4.1.2. The interested reader will find revisions on time series clustering techniques in the works of Liao (2005) and Rani and Sikka (2012).

2.4.1.1 Complete time series clustering

The first subtask consists of evaluating complete sets in a database and grouping them in such a way that similar series are put in the same group. Many authors have proposed different methods for complete time series clustering. The vast majority of these methods are adaptations of traditional clustering methods in such a way that they can deal with time series.

Definition 9 (Complete time series clustering). Considering a distance function $d(X_1, X_2)$ between two series X_1 and X_2 , The clustering of complete series tasks consists on finding the set of C groups $C = \{C_1, \dots, C_b\}$ where $C_i = \{X_k | X_k \in DB\}$ which maximizes the distance between groups and minimizes the variance within the group, i.e., $\forall i_1, i_2, j$ such that $X_{i_1}, X_{i_2} \in C_i, X_j \in C_j$ and $d(X_{i_1}, X_j) \gg d(X_{i_1}, X_{i_2})$.

Two are the main approaches used to adapt traditional clustering algorithms to deal with time series (JAIN; MURTY; FLYNN, 1999). In the first approach, the modification is inside the algorithms. The main modification is associated with the substitution of the distance measure in such a way that they can differentiate time series. The second approach is related to the data. In this approach, data is transformed from time series to a vector of characteristics (characteristic-based methods) or model parameters (model-based methods). Then, the traditional algorithms can be applied without any change. In the following, we present some methods for each approach of adaptation.

- *Methods based on raw data*: These methods do not transform the data but they adapted the traditional methods to deal with time series. Golay *et al.* (1998) applied the *k-means* algorithm to group univariate time series from magnetic resonances to map the activity of the human brain. Windischberger *et al.* (2003) and Dimitriadou *et al.* (2004) have also used other algorithms to group magnetic resonance images. Wijk and Selow (1999) used an agglomerative hierarchical algorithm to group daily, weekly and annual energy consumption series with the objective of finding consumption patterns. Möller-Levet *et al.* (2003a) proposed a clustering algorithm based on *fuzzy c-means* to group small time series that with different sampling intervals. Authors also applied this method with a new similarity measure to cluster DNA microarrays. This measure can distinguish the forms generated by the variation of the amplitude and temporal correlation of the information. Chandrakala and Sekhar (2008) have adapted the DBSCAN method to be applied to multivariate time series. Salvador and Chan (2005) developed a hierarchical method, called Gecko, based on graphs that find clusters in time series in three steps. In the first step, the method finds small subgroups. In the second step, it merges similar groups and then, in the last step, it automatically discovers the best number of groups. This method was used to detect anomalies. Rodrigues, Gama and Pedroso (2008) present an algorithm for hierarchical divisional grouping of time series into data flows.

- *Methods based on features:* These methods consist of converting the raw data into a vector of low-dimensional characteristics and then using the traditional clustering algorithms without any adaptation. This data characterization can bring advantages over the methods that use raw data (without any preprocessing). The main advantages are the reduction of processing time due to possible reductions in data dimensionality and the elimination of noise, which makes clustering easier. Vlachos *et al.* (2003) present an algorithm based on image resolution that iteratively finds groups using several resolutions of series obtained through the Haar transform. First, each element of the data set is decomposed using Haar Wavelet. Then, the k -means algorithm is applied at each level, starting at the coarser resolutions (levels) to the finer levels. The idea of the algorithm is to decrease the amount of data using smaller resolutions. If at the end of each iteration, some data has changed the groups, then the algorithm moves to the next level and begins to analyze finer resolutions, iteratively increasing the amount of data. Otherwise, the algorithm ends. Another feature-based method has been proposed by Guo, Jia and Zhang (2008) and consists of converting raw data into low-dimensional vectors using independent component analysis (HYVÄRINEN; KARHUNEN; OJA, 2004), and then perform clustering using the vectors and k -means. Aghabozorgi *et al.* (2014) proposed a hybrid clustering algorithm based on the shape similarity between each pair of series. First, there is a reduction in the dimensionality of the data using the concept of affinity. The result of this process are subgroups that are clustered using the k -medoids algorithm.
- *Methods based on models:* These methods consider that a model or a mixture of statistical probabilities generate the groups. They consider two time series similar if the characteristics defined by the models to characterize the series are similar. Ramoni, Sebastiani and Cohen (2002) proposed a Bayesian clustering method called BCD. In this method, time series are modeled using Markov chains. Then, a hierarchical agglomerative method is used to group the most similar chains. The same authors also present a method for clustering multivariate series by using modeling of Markov chains. Xiong and Yeung (2002) present a model-based method for grouping univariate ARIMA series. The method assumes that k different ARMA models generate the series and each model represents a cluster. The EM algorithm has been improved to find the number of groups automatically.

2.4.1.2 Subsequence clustering

The second time-series grouping subtask consists of extracting subsequences from one or more series and grouping them in such a way that similar subsequences are in the same group. This subtask is primarily used as the basis of other temporal data mining tasks such as classification (COTOFREI; STOFFEL, 2002), forecasting (SCHITTENKOPF; TIÑO; DORFFNER, 2002), and anomaly detection (YAIRI; KATO; HORI, 2001).

Definition 10 (Subsequence clustering). The subsequences clustering aims to find the set of

groups $C = \{c_1, \dots, c_p\}$ where $c_i = \{S^X | X \in DB\}$ is a set of subsequences that maximizes the distance between groups and minimizes the distance within the group.

Hebrail and Hugueney (2000) divide series into windows without overlapping. The window size is chosen using the discrete Fourier transform to analyze periodic structures of the series. The main limitation of this technique is that if the time series does not have periodic structures, then the windows without overlapping can lose important structures. To try to solve this problem, authors use smaller overlapping windows and then group the results. However, Keogh, Lin and Truppel (2003) demonstrate that using sliding windows to generate subsequences provides insignificant results. The author used the term “insignificant” because the results obtained by the algorithms are independent of the input. By definition of the algorithm k -means, each cluster centroid is the mean of the values of the group. In clustering of subsequences, each centroid is a series whose values are the mean values of the group subsequences. Regardless of the data set, the centroids find sine-derived curves.

In contrast to the work of Keogh, Lin and Truppel (2003), Denton (2004) proposes a clustering algorithm based on density that can find meaningful clusters. Simon, Lee and Verleysen (2006) propose a preprocessing technique that uses delay operator to perform the series unfolding and create regressors that are more informative than sliding windows. The authors used self-organizing maps (SOM) to test the method efficiency to cluster time series. Chen (2007) also propose a solution for clustering by constraining grouping space. Kremer, Gunnemann and Seidl (2010) present an algorithm for the detection of variations in multivariate time series. The series are divided into disjoint intervals of the same size and grouped by using a density-based algorithm. The method tracks clusters over time and obtains the patterns from these groups.

Time series clustering is interesting for many fields of science such as genetics, economics, finance, remote sensing, engineering, geology, and meteorology. In what follows, we cite some examples of applications. Steinbach *et al.* (2003) proposed a method to obtain climatic indexes based on groups of regions on the planet that have similar behaviors. The centroids of each group are time series that summarize the behavior of the ocean and atmosphere in these areas. Alonso *et al.* (2006) proposed a method to group emission series of CO_2 with the objective of creating classes of countries with similar emission. Chouakria and Nagabhushan (2007) have clustered series of CD4 + cell progression response in patients affected by the HIV. Cerra *et al.* (2011) used a measure of similarity between spectral signatures (electromagnetic radiation) and a hierarchical clustering algorithm to categorize spectra of different rocks. Samé *et al.* (2011) used time series of energy consumptions from engines that perform the alteration of tracks in train tracks. The authors applied clustering algorithms to these time series and used the resulting information to find faulty devices.

2.4.2 Time series classification

The classification task consists of categorizing unlabeled time series. The main difference for the clustering task is that, in this case, there is a pre-labeled set of series that is used to train a classifier. For example, a cardiologist can classify series of electrocardiograms if they correspond to healthy hearts or not. The first step is to train a classifier with these labeled series, that learns to differentiate the classes according to the characteristics of the series. This classifier is then used to determine the label of an unknown series.

Definition 11 (Time series classification). Given a set of labels Y , The time series classification task consists of training a classifier C , which is a mapping function of a series X for a label $y \in Y$, i.e., $C : X \rightarrow y, y \in Y$. This mapping function is then used to classify unknown series.

Definition 12 (Linear separability). In the classification context, a data set is said to be linearly separable if it is possible to draw a line (or hyperplane in higher-dimensional Euclidean spaces) that divides data from two distinct classes.

Firstly, we present a review of the main classification methods divided into five categories (KOTSIANTIS, 2007): logical algorithms, perceptron-based algorithms, statistical learning, instanced learning, and support vector machines.

- *Logical Algorithms*: These methods are divided into decision trees and rule-based classifiers. Decision trees are tree structures that classify data by ordering them based on their characteristics. Each node of the tree represents a feature and each branch a value that the node can assume. The leaves represent the classes. Data are sorted from the root node through the tree using their characteristics until they reach a leaf node. The construction of an optimal decision tree is an NP-Complete problem (HYAFIL; RIVEST, 1976). Thus, some heuristics have been proposed to speed up the tree construction. The most popular heuristics are the ID3 (QUINLAN, 1986) and C4.5 (QUINLAN, 1993). Decision trees can be translated into a set of rules by extracting all paths from the root node to each leaf node. But there are other ways of generating these rules. These algorithms are called rule-based classifiers. Among the most well-known algorithms are RIPPER (COHEN, 1995) and PART (FRANK; WITTEN, 1998). We point the interested reader to the work of Safavian and Landgrebe (1991) that presents a review on decision trees.
- *Algorithms based on perceptron*: This class of algorithms uses a binary classifier known as the perceptron (ROSENBLATT, 1962). In the context of neural networks, the perceptron is an artificial neuron. The perceptron is a binary function defined as:

$$f(x) = \begin{cases} 1 & \text{if } w \cdot x + b > 0 \\ 0 & \text{otherwise,} \end{cases} \quad (2.2)$$

where x is an input vector that defines an object of the data set, w is a vector of weights, and b is a constant term known as bias. The perceptron can work in one or multilayer networks. The training process consists of adjusting the weight vector w using objects in the data set until the classification error is zero or very close to zero. The WINNOWER is another example of an algorithm to train a single-layer network (BLUM, 1997). The main problem of the single-layer perceptron is that it can only classify linearly separable data set. The Multilayer Perceptron (MLP) have been created to solve this problem (HAYKIN, 2009). MLPs depend on three fundamental aspects: input vectors, neuron activation functions and the weight connections that form the network architecture. Feed-forward networks process the input signal in only one direction, from the input to the output layer. The main algorithm for training these networks is the backpropagation (RUMELHART; HINTON; WILLIAMS, 1986). In recurrent networks (RNN), such as the Hopfield networks (HOPFIELD, 1988), the signal may go back to previous layers. Another type of network widely used are the radial-based networks (HAYKIN, 2009). These networks use radial basis functions as the activation function. For a review of classification methods based on neural networks, we refer the interested reader to the works of Zhang (2000) and Haykin (2009).

- *Statistical learning*: Statistical methods consider that exists a statistical model that defines the probability of an object belonging to a given class. The simplest algorithm, called the Naive Bayes (RISH, 2001), assumes that the features that define the objects are independent of each other. The algorithm uses the Bayes' theorem to calculate the probabilities of a given object belonging to a class l_i . Thus, the algorithm can be reduced to selecting the class that maximizes:

$$p(C_i) \prod_k p(X_j^k = a_k | l_i), \quad (2.3)$$

where $p(X_j^k = a_k | l_i)$ is the conditional probability that defines the frequency that an attribute X_j has a value a_k in the training set, considering that it belongs to class l_i . The main advantage of this method is the low computational time required in the training process. One disadvantage is that this algorithm assumes that the attributes are independent, which in many cases is not true. This assumption may decrease the accuracy of the method in these cases. A Bayesian network is a directed acyclic graph that represents the characteristics (attributes) of the data by vertices the links represent conditional dependencies between the features. (FRIEDMAN; GEIGER; GOLDSZMIDT, 1997). The absence of edges between two vertices means that these features are conditionally independent. The training process consists of first constructing the network structure and then defining the probabilistic parameters. Given an input object, the user can use inference to find the probabilities of that object belong to each of the classes. The highest probability defines the label of the object. An advantage of these networks is the ease that the human being has in visually understanding the relationship between the attributes. The main disadvantage is the high

computational cost required during the construction of the network. A Bayesian network that relates temporal attributes (time series) is called the dynamic Bayesian network. One well-known form of dynamic Bayesian networks is the Hidden Markov Models (HMM). (MARSLAND, 2009). In these models, each time unit in the system is in a different state, which may be the same as the previous one. In HMMs, the user does not know exactly which state the system is but the output is visible. The main advantage of these models is that the inference is less computationally costly compared to conventional Bayesian networks. Jensen and Nielsen (2007) present a review Bayesian networks.

- *Instance based learning*: Unlike decision trees, artificial neural networks, and Bayesian networks, the instance-based algorithms (IBL) are considered lazy methods because they do not induce or propose a generalization before classifying the data. Instead, IBL performs classification by looking at only the training data. The training phase consists of only extracting and storing the characteristic vectors that represent the data. The best-known instance-based algorithm is k nearest neighbors (k -NN). This algorithm represents each object of the data set as a point in a d -dimensional space, where d is the number of attributes. Given a new object, the algorithm consists of using a distance function to calculate the similarity of this new object to all the others of the dataset. Among the most used functions are the Euclidean distance, Manhattan, Minkowsky, and Mahalanobis (TAN; STEINBACH; KUMAR, 2005). The k -nearest neighbors are the k elements with the shortest distances to the unclassified object. The class of this object is the class with the highest occurrence in the set of k -nearest neighbors. Instance-based algorithms require less computational time in the training phase but need more time during the classification stage. A disadvantage of this method is the dependence of the parameter k and the distance function. If k is too small, noisy data might lead to classification errors. If k is too large, classes with more data will have a greater influence on the classification process, and may also lead to classification errors. Another problem is the high computational cost of space since they need to store a lot of data in memory. Several of these problems were solved using different strategies. For a detailed review of IBL algorithms, we point the interested reader to Aha (1997).
- *Support vector machines (SVM)*: In a linearly separable data set, it is possible to find several lines that divide the data from two classes. However, there is one line that separates the data with the greater margin of separation. The purpose of an SVM (CORTES; VAPNIK, 1995) is to find a line of separation between two classes that maximizes the separation margins. In non-linearly separable sets, a margin of tolerance error, called flexible margins, can be used. Another approach is to make a transformation in the data to make them linearly separable. This transformation consists of representing spatial data using more dimensions. In this case, SVM aims to find the hyperplane that maximizes the separation margins. One SVM cannot solve classification problems involving more than two classes.

However, there are some strategies to deal with this situation (HSU; LIN, 2002). The most common approach consists of training different SVMs for each class. Each SVM knows how to separate its class from the others. In this case, the problem arises of how to know which SVM knows how to recognize a given object. The simplest idea to solve this issue is to choose the SVM with the largest output function. We refer the interested reader for a review of SVM in the work of Burges (1998).

Similar to the task of time series clustering (section 2.4.1.1), There are two approaches for the use of traditional classification algorithms for time series (KOTSIANTIS, 2007). The first modifies the distance functions to compare series and the second one transforms the sequential data into feature vectors or model parameters. The work of (XING; PEI; KEOGH, 2010) presents a detailed review of the methods. We briefly present the main methods for each category as follows:

- *Distance-based methods*: Distance-based classification algorithms, such as k -NN, can be used to classify time series if the distance measure is adapted to differentiate series (Sec. 2.3). The Euclidean distance (Eq. 2.1) is one of the most used (WEI; KEOGH, 2006). (XI *et al.*, 2006) show that the 1-NN classifier (k -NN with $k = 1$) with the DTW presents good results in time series classification. Ratanamahatana and Keogh (2004) propose a method to classify series that uses an optimized version of the DTW measure that decreases the computational time required for the calculation. Prekopcsák and Lemire (2012) Used k -NN with variations of Mahalanobis distance. Silva, Souza and Batista (2013) used recurrence graphs to represent series and calculate the distance between them using the Campana-Keogh distance (CK-1). The authors show that this approach improves accuracy when compared to Euclidean distance and DTW.
- *Feature-based methods*: Classification methods, such as decision trees, use feature vectors that define each object. In this approach, the methods transform a time series into feature vectors and then apply traditional classification methods without any adaptation. Nanopoulos, Alcock and Manolopoulos (2001) used artificial neural networks to classify series. Eads *et al.* (2002) used a genetic algorithm to select features and train an SVM to classify series. Ye and Keogh (2009) introduced the concept of shapelets, which are the subsequences that best represent a class (label). Shapelets can be used to train a classifier and perform classification. The same authors propose a classifier that integrates the process of selecting shapelets with a decision tree. Sugimura and Matsumoto (2011) adapted a statistical text mining measure called TF * IDF for time series and extracted characteristics of the series using a genetic algorithm. These characteristics are used to train a decision tree-based classifier.
- *Model-based methods*: Model-based methods consider that a model or mixture of probability distribution generated the data. One of the simpler models adapted the Naive

Bayes classifier to deal with time series (LEWIS, 1998). Other model-based techniques use other classifiers, such as Bayesian networks (PAVLOVIC; FREY; HUANG, 1999), recurrent neural networks (RNN) (HÜSKEN; STAGGE, 2003), decision trees Hidasi and Gáspár-Papanek (2011) and Bayesian classifiers with Gaussian mixtures (POVINELLI *et al.*, 2004).

In addition to these methods, there are two other approaches for the classification of time series: semi-supervised algorithms and ensembles of classifiers. The semi-supervised algorithms consider that only some elements of the training set are previously labeled. This situation occurs in many real cases where there is little information about the classes. Among the main semi-supervised algorithms are the 1-NN (k -NN with $k = 1$) Wei and Keogh (2006), HMM (ZHONG, 2004) and SVM (WESTON *et al.*, 2005). Ensembles use more than one algorithm in an attempt to minimize their particular drawbacks and maximize the classification performance. Alonso *et al.* (2008) used k -NN, SVM and Naive Bayes. Jović, Brkić and Bogunović (2012) and Deng *et al.* (2013) used decision trees ensembles. Seker *et al.* (2013) used k -NN, SVM and decision tree (C4.5).

The classification of time series has many fields of applications. Yang *et al.* (2005) used classification methods to classify time series of monthly expenses with telecommunications companies to detect users who wish to leave the service and migrate to competitors. Kampouraki, Manis and Nikou (2009) used SVM to classify time series of heartbeats. In this study, it was possible to distinguish electrocardiograms from younger and older people. It was also possible to differentiate healthy people and people with coronary diseases. Chen, Nikolov and Shah (2013) used time series from the activity of a particular subject discussed in Twitter to train a classifier to recognize and predict if a specific subject will become a trending topic.

2.4.3 Time series segmentation

The central purpose of segmentation (or summarization) is to reduce the dimensionality of the series without losing its main features (ESLING; AGON, 2012). The idea is to find the important points that characterize the series and remove the less important ones. Figure 5 illustrates an example of the time series segmentation process. This task can be considered a preprocessing for other data mining tasks or as a trend analysis technique (FU, 2011). Also, it may also be considered a discretization problem.

Definition 13 (Time series segmentation). Given a time series $X = \{x_1, \dots, x_n\}$, the segmentation task consists of constructing a $\bar{X}$ such that $|R(\bar{X}) - X| \leq \varepsilon_r$ where $R(\bar{X})$ is the reconstruction function of $\bar{X}$ for X and ε_r is an arbitrary limiting error.

Several algorithms were proposed to perform the segmentation of series. Keogh *et al.* (1993) categorize the time series segmentation algorithms in on-line or off-line. On-line methods

Figure 5 – An example of time series segmentation. From a time series (a) composed by many values and noise, the objective of the segmentation is to find the most important points of the series and to remove the less important ones. (b) Illustrates the main points of the original series. A low-dimensional approximation that preserves the main features of the original series will be obtained by drawing segments between the main points.

Source: Esling and Agon (2012).

can process new data iteratively and do not require to have the whole series. On the other hand, off-line algorithms always need the entire time series to perform the segmentation. Among the segmentation methods, some of them use piecewise linear approximation (PLA). In summary, these techniques consist of approximating a series X of size n using K discrete segments. Keogh *et al.* (1993) also propose a categorization for algorithms that use PLA which divides the methods into three categories: sliding windows, partitional (top-down) and agglomerative (bottom-up). We briefly present each category as follows.

- *Sliding-window based algorithms*: These methods initially create one small segment that approximate to a subsequence of the series. This segment is then gradually increased until it exceeds an arbitrary threshold error. When it happens, a new segment starts from the endpoint of the previous segment. This process repeats for the next points that were not included in this newest segment (QU; WANG; WANG, 1998; WANG; WANG, 2000; PARK; KIM; CHU, 2001). The advantages of these methods are the simplicity and the on-line approach. The disadvantage is related to the accuracy, generally inferior to other approaches (SHATKAY; ZDONIK, 1996).
- *Partitional algorithms*: In this approach, the method recursively breaks the time series until it reaches a stop criterion. Among the algorithms that use this approach are the work of Li, Yu and Castelli (1998), Park, Lee and Chu (1999) and Lavrenko *et al.* (2000). These methods present good results but demands high computational time.
- *Agglomerative algorithms*: Unlike the partitional approach, agglomerative algorithms create many small segments and iteratively join segments until they reach a stop criterion. Examples of algorithms have been proposed by Keogh and Smyth (1997), Keogh and Pazzani (1998) and Keogh and Pazzani (1999). This methods present better results than the partitioned techniques.

Several other techniques have been proposed for the same purpose using different strategies. Fu, Chung and Ng (2006) proposed a method for segmentation of financial series based on the concept of importance. The more important the data, the sooner this point will be considered a cut-off point in the series. Guralnik and Srivastava (1999) performed the segmentation by detecting points where there is a change in their behavior of the series. The concepts of information theory were also applied to the segmentation of series. Duncan and Bryant (1996) used dynamic programming to find the number and location of ranges in the data. Oliver, Baxter and Wallace (1998) used the idea of minimum message length (MML) and Fitzgibbon, Dowe and Allison (2002) used the concept of minimum expression length (MDL). Xu *et al.* (2013) proposed a hybrid version of MML and MDL to segment series into data streams. For a broader review of segmentation methods, we refer the interested reader to the work of Fu (2011).

Segmentation of series has been applied to speech recognition, geography, forest sciences, finance, biology, physiotherapy, among others. Adami and Hermansky (2003) proposed a method to segment time series of human speech and used them to characterize the speaker and the spoken language. Mithal *et al.* (2012) applied series segmentation techniques to monitor plant coverages. This work was used to detect deforestation in the state of Mato Grosso - Brazil. Si and Yin (2013) proposed and applied a segmentation method to detect points of change in financial series. Omranian *et al.* (2013) presented a network-based method to segment multivariate series extracted from metabolic processes in biological systems and to detect critical events. In physical therapy, the rehabilitation process consists of performing repetitive exercises supervised by a physiotherapist. Lin and Kulic (2014) propose an on-line approach for segmentation and identification of repeated movements that allows the automatic analysis of rehabilitation exercises.

2.4.4 Anomaly detection

The anomaly detection task consists of finding points or subsequences that differ from the pattern. These subsequences are called anomalies or outliers. Figure 6 illustrates an example of anomaly detection. Gupta *et al.* (2013) divide the task of anomaly detection into subtasks according to the data type (Fig. 7): temporal data, data flows, distributed data, spatiotemporal data and networks. In this section, we focus on two types of data: temporal data and temporal networks. We will also present the main algorithms for these kinds of data. For a full and detailed review of all anomaly detection, we refer the interested reader to the works of Chandola, Banerjee and Kumar (2009) and Gupta *et al.* (2013). For a review of statistical methods for outliers detection, we suggest the works of Hawkins (1980) and Rousseeuw and Leroy (1987).

Figure 6 – An example of a periodic series similar to those obtained by electrocardiograms. In this series, it is possible to notice a subsequence that differs from the pattern (anomaly). Anomaly detection consists of detecting points, subsequences or even complete series that differ from the pattern.

Source: Esling and Agon (2012).

Figure 7 – The outlier detection process in temporal data can be subdivided according to the data type: temporal data, data flows, distributed data, spatiotemporal data, and networks.

Source: Adapted from Gupta *et al.* (2013).

2.4.4.1 Outliers detection in time series

Outliers detection in time series can be subdivided in the detection of anomalous series in a database or in finding anomalous subsequences in only one series. These two approaches will be presented below:

- Outliers detection in database:

- *Direct outliers detection*: The goal of the direct outliers detection is to find all the anomalous series in a database. In this approach, the detecting methods consider that

the majority of the series of a database is normal and only a few ones are anomalous. To solve this task, some methods use models to represent the series and those that are different from the model are considered anomalous. These models can be supervised and unsupervised. In supervised methods use a classifier to identify anomalous series. Some of the supervised methods used for anomaly detection are the neural networks (ART) (DASGUPTA; NINO, 2000), support vector machines (SVM) (LI; HAN; KIM, 2006), classifiers based on rules (LI *et al.*, 2007a) and hidden Markov models (HMM) (GAO; MA; YANG, 2002). In unsupervised methods, the idea is to group similar series and measure the anomaly degree of a given series by calculating the distance from this series to the centroid of the nearest group. In this approach, several clustering algorithms were used. Among them are the k -means (NAIRAC *et al.*, 1999), k -medoids (BUDALAKOTI *et al.*, 2009), k -NN (CHANDOLA; MITHAL; KUMAR, 2008), single-linkage (PORTNOY; ESKIN; STOLFO, 2001), SVM (MA; PERKINS, 2003b), and Self-organizing maps (SOM) (GONZÁLEZ; DASGUPTA, 2003).

- *Outliers detection in sliding windows*: This task consists of finding unusual time windows in a series database and consequently detecting anomalous series. The anomaly degree of a series is the anomaly degree of all the windows that compose the series. This approach yields better results than the direct detection of outliers. However, many methods require informing the size of the sliding window. One way to implement this solution is to store all windows of size w from all series in a database DB along with their respective frequencies. Given the windows frequencies and an unknown series, extract all windows of size w from that series and check for each window its incidence in the DB database. For each window that does not occur in DB , an anomaly score is given to the series. Series with many points can be considered anomalous. Some techniques use this approach by considering the exact match between time windows (GHOSH; SCHWARTZBARD; SCHATZ, 1999; CABRERA; LEWIS; MEHRA, 2001) and others consider the approximate correspondence (LANE; BRODLEY, 1997; LANE; BRODLEY, 1999).
- *Outliers detection in subsequences*: Given a time series database, this task consists of finding an anomalous subsequence taking into account the frequency that this subsequence occurs in the database. Among the methods used to solve this task are suffix trees (KEOGH; LONARDI; CHIU, 2002; LIN *et al.*, 2007) and interpolated Markov models (IMM) (GWADERA; ATALLAH; SZPANKOWSKI, 2003).
- Outliers detection in just one time series:
 - *Anomalous points detection*: It consists of detecting anomalous points in only one time series. One approach to solving this problem uses forecasting models. For each point x of a series X , it verifies the value predicted by the prediction model x_p and

calculates the difference to the observed real value. This difference can be used as an anomaly score (BASU; MECKESHEIMER, 2007). Many classifiers were used with this approach, including multilayer perceptron networks (MLP) (HILL; MINSKER, 2010), support vector regression (SVR) (MA; PERKINS, 2003a) and artificial neural networks (SILVESTRI *et al.*, 1994).

- *Anomalous Subsequence detection*: This subtask consists of detecting anomalous subsequences in just one time series. Given a series X of size n and a subsequence $S^X = \{x_t, x_{t+1}, \dots, x_m\}$ of X size $m < n$ starting in time x_t , the subsequence S^X is considered anomalous if its distance to the nearest corresponding subsequence without overlapping is greatest among all pairs of subsequences of X (KEOGH *et al.*, 2006). To calculate this greater distance using brute force, consider all subsequences $S^X \in X$ of size m and compute the distance for all other subsequences $S'^X \in X$ not correspondent to S^X . Comparison between subsequences can be optimized by pruning unnecessary subsequences. Some algorithms use hash table (WEI; KEOGH; XI, 2006) and Haar wavelet with tree structure (BU *et al.*, 2007) to do this optimization.

2.4.4.2 Outliers detection in temporal networks

Many dynamic systems can be modeled using complex networks. In this approach, it is possible to study the behavior of dynamic systems without actually studying the dynamics that govern it. Once they are dynamic systems, the networks that represent them are also dynamic. Thus, vertices and edges can appear or disappear according to the evolution of the modeled system. These dynamic networks, also called temporal networks (HOLME; SARAMÄKI, 2012), can be studied by the mining of temporal data. The most common way of studying is to take “snapshots” from the topological state of the network at any given time frequency and then obtain a series of “snapshots” from the network. From this sequence of networks it is possible to detect anomalous states of the network. We present some approaches for outliers detection in temporal networks as follows:

- *Outliers detection using graph similarity*: Given a finite time series composed of topological states of a network, this task consists in detecting abnormal states. One of the primary approaches uses distance functions between graphs to compare adjacent graphs in time and creates a time series of differences. Pincombe (2005) presents several distance functions between graphs, among them are the editing distance (Levenshtein) for graphs, modal distance, distance using diameter, entropy, spectral distance, etc. Another distance function between graphs evaluates the variation of the shortest path between two vertices over time. Gupta, Aggarwal and Han (2011) studied the problem of finding the pair of vertices that has the greatest variation of the minimum distance in two consecutive instants of time. Pincombe (2005) uses an ARMA model to represent the normal changes in a series. Residuals that exceed a certain limiting value are considered anomalous.

- *On-line outlier detection in graphs*: These methods search for anomalous states in graph streams. Idé and Kashima (2004) proposed a method for anomaly detection in streams using eigenvectors. The main eigenvector of the weight matrix in time t is called the active vector $u(t)$. Within a h range, the active vectors are stored in a matrix $U(t) = [u(t), u(t+1), \dots, u(t+h-1)]$. It is possible to define from $U(t)$ the typical behavior pattern of the series. From this pattern, the method gets a typical standard vector and use it to compare to the active vector in time t . The angle between these two vectors defines the anomaly degree of the active vector in the time t . The algorithm automatically determines the threshold degree that defines an anomaly. Aggarwal, Zhao and Yu (2011) proposed another algorithm to detect anomalies in large graph stream. This algorithm tries to find unusual relations (edges) in large temporal networks. This algorithm uses models that summarize the structure of the graph in the data flow. The algorithm maintains the models using a random sampling approach that compresses the structure of the graph representing the flow. These models define the edge generation probability. The geometric mean of the likelihood adjustments of its edges defines the probability adjustment of a graph. For each graph that arrives in the flow, the method calculates the probability adjustment. Graphs whose probability adjustments are t standard deviations less than the average probability of all graphs are considered outliers.
- *Outliers detection based on community detection*: Given a finite time series composed of topological states of a network, this task consists of finding anomalous variations in the communities. In graphs sequences, vertices that belong to the same community tend to have a similar evolution. However, it does not happen for some vertices of the community. The purpose of this task is to find these anomalous vertices, also called by Gupta *et al.* (2012) Evolutionary Community Outliers (ECOOutliers). The same authors proposed a method to simultaneously detect corresponding communities (concept similar to Definition 8) over time and detect ECOOutliers. Gupta *et al.* (2011) presented a set of measures that can be used to detect community-based outliers.

The detection of outliers in time series has several applications in several fields. One interesting application is the invader detection in computer networks (SEQUEIRA; ZAKI, 2002; ANGIULLI; FASSETTI, 2007). Hill and Minsker (2010) used outliers detection to find measurement errors in wind speed data. Keogh, Lin and Fu (2005) used outliers detection to find anomalous subsequences in electrocardiograms. Yankov, Keogh and Rebbapragada (2007) used data from historical searches in The Microsoft Network (MSN) to detect upcoming events such as Easter and Christmas. Ge *et al.* (2010) applied anomaly detection methods to trajectories data. They found anomalies in traffic, such as cars driving in the opposite direction or people crossing the street in forbidden places.

Figure 8 – Motif detection consists of finding subsequences in time series that occur frequently. These subsequences are called motifs. This figure illustrates two examples of motifs in different colors. Motifs can have different sizes and may be overlapping.

Source: Esling and Agon (2012).

2.4.5 Motifs detection

In time series, motifs are subsequences that appear frequently. These motifs can have several sizes and may be overlapping. Figure 8 illustrates an example of a time series with motifs. The naive detection approach consists of verifying all the possible subsequences. However, it is a combinatorial problem and computationally infeasible for long time series. This task becomes even more complicated when in data streams.

Definition 14 (Motifs detection). Given a time series $X = \{x_1, \dots, x_n\}$, the motif detection consists of finding the subsequences S^X that repeatedly occur in X .

Motif detection can be used as an auxiliary technique for other temporal data mining tasks such as segmentation, clustering, and classification (MALETZKE *et al.*, 2014). Lin *et al.* (2002) introduced the concept of motifs in time series. In this work, the authors present an algorithm for motif detection that first reduces the series dimensionality through discretization. Then, the authors use a distance function similar to the Euclidean distance, but with lower computational cost. The method uses this distance function to search for repeating subsequences. This approach presents two problems: low scalability and low efficiency when the series presents noises. Chiu, Keogh and Lonardi (2003) presented a probabilistic algorithm to solve these two problems.

Most methods search for motifs with a fixed size (CHIU; KEOGH; LONARDI, 2003; FERREIRA *et al.*, 2006; MUEEN *et al.*, 2009; YANKOV *et al.*, 2007). Since motifs can have multiple sizes, most techniques use an arbitrary parameter that defines the motif size to be fetched. A simple strategy to find motifs of various sizes is to execute the algorithms with different this parameter. However, this strategy may have a high computational cost due to successive repetitions. Mueen (2013) proposes an exact method for enumerating motifs in series that is faster than executing an algorithm several times. The algorithm searcher for all sizes of motifs and returns only the largest ones. For more methods, we refer the interested reader to the works of Minnen *et al.* (2007b), Tanaka, Iwamoto and Uehara (2005) and Tang and Liao (2008).

Several other techniques use different approaches to find recurring subsequences in series (MUEEN; KEOGH; BIGDELY-SHAMLO, 2009). Minnen *et al.* (2007a) define multidimensional and sub-dimensional motifs. Some authors presented methods to detect both types of motifs (MINNEN *et al.*, 2007b; MINNEN *et al.*, 2007a; TANAKA; IWAMOTO; UEHARA, 2005). There are also algorithms that find motifs that appear more than a arbitrary number of times defined (CHIU; KEOGH; LONARDI, 2003; FERREIRA *et al.*, 2006) or those that only search a pair of closer subsequences (MUEEN *et al.*, 2009). Mohammad and Nishida (2009) presented the idea of motifs with constraints. Motifs can also be restricted at a lower limiting distance (CHIU; KEOGH; LONARDI, 2003; FERREIRA *et al.*, 2006; YANKOV *et al.*, 2007) or restricted to a minimum density (MINNEN *et al.*, 2007b). Algorithms can also search for approximate or exact motifs (MUEEN *et al.*, 2009; NARANG; BHATTACHERJEE, 2010; MOHAMMAD; NISHIDA, 2014).

Motifs detection has been applied in several fields. The study of renewable resources and energy production (such as the wind) requires a short-term forecast (up to 6 hours) and planning how much energy will be produced (KAMATH; FAN, 2012). Motif detection methods can find patterns in series and assist in the production planning. Brown *et al.* (2013) filmed and extracted series from the motion of the nematode *Caenorhabditis elegans*. The authors applied motifs detection techniques to construct a dictionary of patterns of the animal behavior. Sensors are used to monitor the structure of bridges on highways that suffer from the influence of cars and weather conditions. Vespier, Nijssen and Knobbe (2013) proposed and applied a motif detection method to find patterns in these time series from sensors. Small, medium and large scale motifs represent the influence of cars, congestion and daily temperature variation on the bridge respectively. Mueen (2014) presents other application of motif detection techniques in time series.

2.4.6 Forecasting

The major objective of time series analysis is forecasting. In temporal data mining, the forecasting task (Fig. 9) can be seen as a regression problem. In classification problems, the expected value for each entry is a discrete value (label). On the other hand, in regression problems, the expected value is a real value. In this way, a time series can be converted into a training set that is used to train a model to make regressions.

Definition 15 (Time series forecasting). Given a time series $X = \{x_1, \dots, x_n\}$, the forecasting task consists of predicting the next h values of the series $\{x_{n+1}, \dots, x_{n+h}\}$.

Among the most used regression methods are the artificial neural networks (ZHANG; PATUWO; HU, 1998), support vector machines (SVM) (SAPANKEVYCH; SANKAR, 2009), self-organizing maps (SOM) (BARRETO, 2007), decision trees (CART) (BREIMAN *et al.*,

1984), Bayesian networks (NEAPOLITAN, 2003), Gaussian processes (RASMUSSEN; WILLIAMS, 2005), and the k -NN (ALTMAN, 1992). We briefly presented describe them below.

Figure 9 – An example of time series forecasting. (a) Input time series. (b) The purpose of the forecast is to predict the next h values (red line) of a series. (c) The higher the h , the harder the forecast.

Source: Esling and Agon (2012).

The k -nearest neighbors approach (k -NN) was initially proposed by Lorenz (1969) for the weather forecast. Considering a time series $X = \{x_1, \dots, x_n\}$ in which it is desired to predict the value of the next element x_{n+1} . First, the authors construct a matrix of time windows W^X of size w , where w is defined by the user. The second step consists in finding in W^X the most similar window of time ($k = 1$) the window that one wishes to predict, i.e., the one more similar to the last subsequence of the series: $\{x_{n-w+1}, \dots, x_{n-1}, x_n\}$. A distance function is required (Sec. 2.3) to calculate the similarity between subsequences. Considering that the most similar window is $W_j^X = \{x_j, x_{j+1}, \dots, x_{j+w-1}\}$ such that $1 \leq j \leq n - w + 1$ and $j \in \mathbb{N}$, the subsequent value after the window is observed, i.e., x_{j+w} . If this pattern has already occurred in the past, then there is a high probability that this value will occur again. Therefore, x_{j+w} is the result of the forecast. In the initial proposal of Lorenz (1969), only the nearest neighbor was observed ($k = 1$). The main extension of this work was to consider the k -nearest neighbors (ALTMAN, 1992).

Regardless of the supervised algorithm, there are some ways to train the algorithm to do regressions. Bontempi, Taieb and Borgne (2013) point out two main strategies that we present as follows.

- *Iterative strategy*: Given a time series $X = \{x_1, \dots, x_n\}$, The iterative (or recursive) strategy (SORJAMAA *et al.*, 2007; HERRERA *et al.*, 2007) consists of using the sliding windows of the matrix W^X as training set. For each window, the desired output value is the value that follows the time window. After the training process, the forecasting process consists of presenting as the model entry the last window of the series: $\{x_{n-w+1}, \dots, x_{n-1}, x_n\}$. The output is the prediction of the next element of the series: x_{n+1} . Then, it is assumed that x_{n+1} actually occurred and the new last window is used as input: $\{x_{n-w+2}, \dots, x_{n-1}, x_n, x_{n+1}\}$ to predict the element x_{n+2} . This iterative process is repeated until the element x_h is predicted.

Example 1 (Iterative forecasting). Consider a time series $X = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7\}$ and that the user wants to forecast the element x_{10} ($h = 3$). We describe below the training data set and the forecasting processes using the iterative approach:

Train		iteration	Forecast	
input	expected		input	output
$\{x_1, x_2, x_3\}$	x_4	1	$\{x_5, x_6, x_7\}$	x_8
$\{x_2, x_3, x_4\}$	x_5	2	$\{x_6, x_7, x_8\}$	x_9
$\{x_3, x_4, x_5\}$	x_6	3	$\{x_7, x_8, x_9\}$	x_{10}
$\{x_4, x_5, x_6\}$	x_7			

During the training process, the model uses each input and the correspondent expected values in the training data set. After this process, the regression model iteratively forecasts the elements x_8 , x_9 and x_{10} using the input time windows.

- *Direct strategy*: In the direct approach, the regression model directly predicts the desired value (CHENG *et al.*, 2006; HAMZAÇEBİ; AKAY; KUTAY, 2009). Different from the iterative approach, it is not necessary to make successive forecasts. However, if the problem involves the prediction of more than one value, other models are required. For the first model, the training data are the sliding windows of W^X of size w and the expected values for each of these windows are the next value after the window. For the second model, the training data are also the W^X windows, but the expected values for each window are the second value after the window. The same idea works for other values of h .

Example 2 (Direct forecasting). Consider a time series $X = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7\}$ and that the user wants to forecast the element x_{10} ($h = 3$). We describe below the training data set with three inputs ($w = 3$) and the forecasting processes using the direct approach:

Train		iteration	Forecast	
input	expected		input	output
$\{x_1, x_2, x_3\}$	x_6	1	$\{x_5, x_6, x_7\}$	x_{10}
$\{x_2, x_3, x_4\}$	x_7			

Note that in this case, the prediction of the value x_{10} is done directly. However, the intermediate values x_8 and x_9 were not predicted. If it is necessary to predict them, then it will be necessary to train two other prediction models, one for x_8 and another for x_9 .

With these strategies, it is possible to adapt classification algorithms to perform forecasts on time series. Lim, Loh and Shih (2000), Ahmed *et al.* (2010) present a comparison between the iterative and direct approaches. For a review on statistical forecasting methods, we refer the interested read to the works of Box, Reinsel and Jenkins (1994), Hamilton (1994), Shumway and Stoffer (2011).

The task with the greatest number of applications is probably the time series forecast. Forecasting techniques have been applied in economics, geography, climatology, medicine, games, telecommunications and several other areas. We present some application examples as follows. Rasouli, Hsieh and Cannon (2012) used Bayesian networks, SVMs and Gaussian processes to predict flow in a watershed in Canada. Reyes, Morales-Esteban and Martínez-Álvarez (2013) used artificial neural networks to predict earthquakes in Chile. Mateo *et al.* (2013) used time series prediction methods to forecast the temperature inside a building and used this information to manage energy consumption. Arias, Arratia and Xuriguera (2014) used indexes to measure interest in a particular subject over time on Twitter. Authors used these indexes in conjunction with other series to improve stock market forecasting and movie sales spending.

2.5 Final remarks

In this chapter, we presented the basic concepts related to time series and main tasks involving time series data mining. The concepts here presented are fundamental for the understanding of the methods and results discussed in the next chapters. In the next chapter, we will review the other main topic related to this thesis: complex networks.

COMPLEX NETWORKS

3.1 Introduction

The study of complex networks has origins in discrete mathematics and graph theory. The famous problem of the six bridges of Königsberg proposed by Leonhard Euler in 1735 is considered one of the first works in the study of graphs. Over the years, graph theory has consolidated itself as an important tool in the study of networks. However, with the development of computers, larger and larger databases were collected, stored and processed. The use of traditional graph theory techniques is unfeasible when dealing with large-scale networks because they seek optimal solutions and consequently require high computational time.

Another motivation for the study of complex networks comes from the area of complex systems. So far, many researchers tried to understand how each individual of a system behaves to understand the collective dynamics. However, it is easy to realize that this reductionism approach has flaws. For example, understanding how the neurons work does not explain consciousness or memory. The interaction of the elements that form the system creates complex behaviors that can not be described by studying each element by itself. As described by Mitchell (2009), many scientists moved beyond the reductionism and started to believe that “the whole is more than the sum of the parts”. One important tool that allowed researchers to study these complex systems is the network theory. Researchers from different areas have been using network models to investigate the dynamics of complex systems. The main advantage of this approach is to be able to represent various systems in a unified form, making it possible to find common characteristics and properties.

We initially present the basic concepts of graph theory used in this thesis (DIESTEL, 2005).

Definition 16 (Graph). A graph G is defined as a pair (V, E) , where V is a finite and not empty set composed by n vertices and E is a set of m edges that connects vertices in such a way that

$$E \subseteq \{(u, v) \mid u, v \in V\}.$$

- **Undirected Graph:** In a non-directed graph, the existence of an edge (i, j) implies that (j, i) also exists, i.e., the relationship E is said symmetric: $\forall (i, j) \in E \implies (j, i) \in E$.
- **Directed Graph (digraph):** In a directed graph, the existence of an edge (i, j) does not imply that (j, i) also exists. In this case, the directed edge (arc) (i, j) means that the vertex i is connected to the vertices j , but the reciprocal is not true, that is $(i, j) \neq (j, i)$.

Definition 17 (Weighted Graph). A weighted graph G is defined as a triple (V, E, W) , where V and E are the sets of vertices and edges respectively, and W is an array of size $n \times n$ of values (weights) associated to each of the edges of the graph.

The main form to computationally represent a graph is through the adjacency matrix A of size $n \times n$, where n is the number of vertices of the network. For every pair of vertices i and j of network, if there exists an edge $(i, j) \in E$ connecting them, then the position $A_{ij} = 1$, otherwise $A_{ij} = 0$. In weighted graphs, each position ij of the matrix can store the weight of the edge (i, j) , i.e., $A_{ij} = w_{ij}$ if $\exists (i, j) \in E$.

Definition 18 (Loop). A loop is an edge (v, v) that connects a vertex v to itself.

Definition 19 (Multiple Edges). Given two vertices i and v , if there exists more than one edge (i, v) connecting these vertices, they are called multiple edges.

Definition 20 (Simple Graph). A simple graph is an undirected graph that does not have multiple edges or loops.

In the remainder of this work, the term “graph” will be used, for simplicity, to refer to a simple graph. When this is not the case, its characteristics will be specified. In addition, the terms “graph” and “network” will be considered similar concepts.

Definition 21 (Adjacent Vertices). Two vertices i and v are adjacent (or neighbors) if there exists an edge $(i, v) \in E$ connecting them. An edge (i, v) that connects two vertices i and v is said incident on the vertices i and v .

Definition 22 (Neighborhood). The neighborhood N_i of the vertex i is defined as the set of the adjacent vertices (neighbors) to the vertex i .

Definition 23 (Vertex Degree). The degree k_i of a vertex i is the number of neighboring vertices of i , i.e., the neighborhood cardinality: $|N_i|$.

Definition 24 (Walk). A walk W in a graph is a sequence of adjacent vertices $W = \{(v_1, v_2), (v_2, v_3), \dots, (v_{k-1}, v_k), (v_k, v_{k+1})\}$. The length of this walk is given by the number of elements (cardinality) of $|W|$.

Definition 25 (Path). A P path is a walk where no vertices and no edges are repeated.

Definition 26 (Cycle). A cycle is a path where the first and the last vertices are the same.

Definition 27 (Vertices distance). Given two vertices i and v , the distance $d(i, v)$ between them is defined as the length of the shortest path between them. If there is no path between i and v , then $d(i, v) = \infty$. The distance from a vertex v to itself is $d(v, v) = 0$.

Definition 28 (Subgraph). Given two graphs $G = (V, E)$ and $G' = (V', E')$, G' is a subgraph of G if, and only if, $V' \subseteq V$ and $E' \subseteq E$.

Definition 29 (Component). A component is a subgraph in which all pairs of vertices are connected to each other by a path, but none of them are connected to any other vertex of the graph.

Definition 30 (Connected Graph). A connected graph is a graph which there is a path between all the pairs of vertices. A connected graph has only one component.

Definition 31 (Regular Graph). In a regular graph, all the vertices of the network have the same degree.

Definition 32 (Complete Graph). In a complete graph, all n vertices are connected to each other. In this case, the number of edges of the graph is $m = n(n - 1)/2$.

Definition 33 (Clique). Considering a graph G , a click is a complete subgraph of G .

Definition 34 (Tree). A tree is a connected graph without cycles and $m = n - 1$ edges.

Definition 35 (Linear Graph). A linear graph, also called a path graph, is a tree where all vertices have a maximum degree equal to 2.

Definition 36 (Null Graph). A null graph is a graph with no edges, i.e., $m = 0$.

3.2 Complex networks models

Studying the network topology is important because its structure affects the network functionality (STROGATZ, 2001). Some researchers proposed network models that present some features observed in real networks. Three network models are the most studied. We describe them as follows.

3.2.1 Random networks

Erdős and Rényi (1959) proposed a model capable of generating random networks. Considering initially a network composed by n vertices totally disconnected ($m = 0$), the generation process consists in creating random edges given an arbitrary probability $p \in [0, 1]$. This model

disregards loops and multiple edges. At the end of the process, the distribution of the number of connections (degree) follows a Poisson distribution. This model has been extensively studied in the literature. However, most of the real networks do not follow this model because the connections are not randomly formed.

3.2.2 Small-world networks

Watts and Strogatz (1998) proposed a model that generates networks with some properties observed in social networks. One of these properties is the transitivity (RAPOPORT, 1953). Given three people A , B and C , considering that A knows B and B knows C , so there is a high probability of A also knowing C . This property makes the network to have subgroups of highly connected vertices (communities) and a high clustering coefficient (c). Another property is the small-world effect. In social networks, the average shortest paths (l) among all pairs of people is small. It happens because people are separated from each other by a small number of people. To generate networks with this properties, the model initially considers a regular network where the vertices are positioned under a ring equally spaced. Each vertex is connected to the nearest k neighbors on each side of the ring, that is, the $2k$ neighbors. Then each edge is randomly reconnected to two other pairs of any vertices with probability $p \in [0, 1]$. When p is small, a network is generated with high c and low l . Figure 10 illustrates how the model proposed by Watts and Strogatz (1998) generates networks with the small-world property

Figure 10 – The process of transforming a regular network into a random network without increasing vertices or edges. The initial regular network (left) has 20 vertices, each of them connects to the four closest vertices. The network generation process was repeated twice for different values of p . The small-world network (center) was generated for small values of p . When $p \rightarrow 1$, the result is a random network (right).

Source: Watts and Strogatz (1998).

3.2.3 Scale-free networks

Barabasi and Albert (1999) demonstrated that several real networks have a topology formed by many vertices with few connections and few vertices with many connections (hubs). They also observed that the degree distribution of these networks follows a power law $p(k) \sim k^{-\lambda}$, where λ is a constant that varies between 2 and 3. In random networks, the degree distribution follows a Poisson and the mean degree $\langle k \rangle$ indicates the scale of the network, since a randomly selected vertex will have degree $k = \langle k \rangle \pm \langle k \rangle^{\frac{1}{2}}$. On the other hand, in networks studied by Barabasi and Albert (1999), the mean degree can not be used as a scale to predict the degree of a vertex taken at random. Therefore, these networks are called free-scale networks.

3.3 Complex network measures

After modeling the interested domain using networks, it is common to exam the topological features of these networks. Another form to study is to evaluate the modifications in the topology over time or to classify different networks. For all of these branches of study, it is fundamental to use measures to quantify properties of the network. These measures can be used to evaluate, characterize and classify networks. Costa *et al.* (2007) define several measures and some of them will be presented bellow.

- *Average degree $\langle k \rangle$* : Average number of vertices connections. Considering A the adjacency matrix of the network, the degree of a vertex i can be defined as $k_i = \sum_{j=1}^n A_{ij}$. The average degree is defined as:

$$\langle k \rangle = \frac{1}{n} \sum_{j=1}^n A_{ij}. \quad (3.1)$$

- *Degree distribution $p(k)$* : Distribution of degree probability of all vertices of the network. Considering that there are n_k vertices of degree k , then $p(k) = n_k/n$. Considering $p(k)$ of all vertices ($k = \{0, \dots, n-1\}$), then we have the distribution of degrees. This distribution brings global information about the connection of the vertices.
- *Average shortest paths l* : The mean of the shortest paths between all the vertex pairs is defined by:

$$l = \frac{1}{n(n-1)} \sum_{i \neq j} d(i, j), \quad (3.2)$$

Where $d(i, j)$ is the distance between vertices i and j . It is an important measure of information flow in the network. For example, considering that we want to transmit information from a vertex i to a randomly chosen vertex j , if l is small, then $d(i, j)$ is also small in the average. Therefore, the information will be transmitted quickly on average.

- *Clustering coefficient c* : This measure is used to verify the existence of highly connected subgroups in the network. More specifically, c measures the transitivity of the network, i.e., the fraction of the cycles of size 3 (N_Δ) among all paths of size 3 (N_3):

$$c = \frac{3N_\Delta}{N_3}. \quad (3.3)$$

- *Betweenness centrality B_u* : The greater the number of paths that pass through a certain vertex or edge, the greater its importance in the network. In this way, it is possible to measure the importance of a vertex (or edge) u by its centrality B_u , defined as:

$$B_u = \sum_{ij} \frac{\sigma(i, u, j)}{\sigma(i, j)}, \quad (3.4)$$

where $\sigma(i, j)$ is the total number of shortest paths between the vertices i and j , and $\sigma(i, u, j)$ is the number of shortest paths between the vertices i and j that pass through the vertex u .

- *Assortativity r* : Assortativity is the Pearson correlation of the vertices degrees of the network (Eq. 3.5). It measures the tendency of vertices with high degree to connect with other vertices also of with high degree ($r = +1$) or to connect with lower degree vertices ($r = -1$). Assortativity equal to 0 means that no correlation exists.

$$r = \frac{\frac{1}{m} \sum_{j>i} k_i k_j A_{ij} - \left[\frac{1}{m} \sum_{j>i} \frac{1}{2} (k_i + k_j) A_{ij} \right]^2}{\frac{1}{m} \sum_{j>i} \frac{1}{2} (k_i^2 + k_j^2) A_{ij} - \left[\frac{1}{m} \sum_{j>i} \frac{1}{2} (k_i + k_j) A_{ij} \right]^2} \quad (3.5)$$

- *Modularity Q* : The modularity measure was proposed by Clauset, Newman and Moore (2004) and measures how good a given network partition is. A partition is a division of the network into groups where each vertex belongs to a group. This measure is used to find highly connected groups (or communities) in the network. Networks with low modularity (close to 0) represent random networks whereas high values (the authors suggest values above 0.3) indicate the presence of communities. The modularity can be calculated by the equation:

$$Q = \frac{1}{2m} \sum_{i,j} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(c_i, c_j), \quad (3.6)$$

where c_i and c_j are communities whose vertices i and j are inserted respectively, and $\delta(c_i, c_j)$ is the Kronecker Delta function that has output 1 if the communities c_i and c_j are equal, or 0 otherwise.

3.4 Community detection

One of the main properties of real networks is the presence of a large number of edges connecting vertices of a subgroup, but few edges connecting that subgroup to other subgroups. Figure 11 shows an example of a network with community structures. Several real networks present these group structures, called communities. In social networks, for example, people tend to form groups at work or in the family (DONETTI; MUÑOZ, 2004). Other examples include scientific collaboration networks (GIRVAN; NEWMAN, 2002), interaction networks between proteins (SPIRIN; MIRNY, 2003), web pages (FLAKE *et al.*, 2002) and food chains (KRAUSE *et al.*, 2003).

Figure 11 – An example of a network with three communities represented by colors. Inside each community, the vertices are highly connected. On the other hand, there are few connections between vertices of different communities.

Source: Elaborated by the author.

Definition 37 (Community Detection). Given a network $G(V, E)$, the process of community detection consists of searching for a set P of groups of densely connected vertices, called communities, using just the topological characteristics, i.e., vertices and edges. This set is defined as $P = \{P_1, \dots, P_b\}$, where $1 \leq b \leq n$ and $P_i = \{v \mid v \in V\}$ is a nonempty set of vertices. Every vertex of G can belong to just one community, i.e., $P_i \cap P_j = \emptyset$ for every pair of communities.

The community detection task in networks is similar to the clustering task in data mining. In fact, it is possible to use the traditional clustering algorithms presented in Sec. 2.4 to perform community detection. However, most of these traditional clustering methods do not take into account the relationship between the vertices during the clustering process. The objective of the community detection algorithms is to use only the topological structure of the networks to group the vertices. Many of these algorithms can find hierarchies of communities, similar to traditional hierarchical clustering methods.

We present below the main approaches used for the community detection as well as their main algorithms (FORTUNATO, 2010).

- *Divisive algorithms*: It consists of removing successive edges from the network until each group forms a disconnected component. The main idea of this approach is to find the edges that connect different groups and remove them. Considering that this removal process is iterative, this approach can find top-down group hierarchies, usually represented by a dendrogram (Fig. 4). The best-known algorithm was proposed by Girvan and Newman (2002) and we summarized it in the following four steps: (1) Calculate the centrality betweenness (Eq. 3.4) of each edge of the network; (2) Remove the edge with greater centrality; (3) Recalculate the centrality for all edges; (4) Return to step 2. In its first version, the algorithm does not return which is the best partition of the hierarchy. The algorithm started using the modularity measure (Eq. 3.6) to find the best partition just in a later version (NEWMAN; GIRVAN, 2004). Although it presents good results, this method consumes high computational time due to successive calculations of centrality. In the worst case, it runs in $O(m^2n)$ time, making its application impossible in larger networks. Subsequent improvements in the algorithm were proposed to reduce the computational time (WILKINSON; HUBERMAN, 2004; RATTIGAN; MAIER; JENSEN, 2007). Another algorithm that uses this approach was proposed by Radicchi *et al.* (2004). This algorithm is based on the idea that within the communities there is a large number of cycles. On the other hand, edges that connect different groups participate in a few cycles. Considering these observations, a measure was introduced to calculate the number of cycles that pass through each edge. Edges with small values for this measure are more likely to be edges between groups.
- *Spectral algorithms*: Spectral properties of graphs can be used to find partitions in the network. The Laplacian matrix L of a graph can be obtained by means of the following function: $L = D - A$, where A is the adjacency matrix and D is a diagonal matrix whose element $D_{ii} = k_i$ stores the degree of the vertex i . Thus, it is possible to use the values of the eigenvectors as coordinates of a plane M -dimensional, where M is the number of eigenvectors used to represent the vertices of the network. When plotted, the vertices that belong to the same communities are located close to each other. Donetti and Muñoz (2004) applied a hierarchical clustering algorithm to group these points together. The algorithms of Capocci *et al.* (2005) and Yang and Liu (2008) also use this approach.
- *Dynamic Algorithms*: Random walks can be used to find communities (QUILES *et al.*, 2008). If a network has a community structure, then a particle that walks randomly over the network is more likely to remain within the communities, since there are more edges within the group than edges connecting groups. Zhou (2003) proposes a method that uses random walks to find the distance between pairs of vertices. This distance $d(i, j)$ between two vertices i and j is calculated by the average number of edges that a Brownian particle must travel to reach j from i . Closer vertices are more likely to belong to the same community. Due to successive distance calculations, this method performs computational

time detection $O(n^3)$. Another dynamic approach uses oscillator synchronization. In this case, each vertex is represented by an oscillator that initially pulses randomly. The oscillators begin to synchronize their pulses from the interaction with their neighbors. When oscillators reach a state of synchronization, their pulses are performed at the same time. Due to the high density of edges within the community, oscillators from the same community tend to synchronize first. Arenas, Díaz-Guilera and Pérez-Vicente (2006), Boccaletti *et al.* (2007) and Zhao, Cupertino and Jr. (2008) also used this approach.

- *Methods based on statistical inference:* Clustering in graphs can be treated as a statistical inference problem. In this case, the idea is to deduce hypotheses models from the connections between the vertices of the graph and make inferences from these hypotheses. One of the most commonly used methods is Bayesian inference (MACKAY, 2003). Bayesian inference uses real observations (evidence) to estimate the probability of a given hypothesis being true. In community detection, the evidences are represented by the graph structure (HOFMAN; WIGGINS, 2008). Other types of inference have also been used, e.g., block-modeling (REICHARDT; WHITE, 2007) and information theory (ZIV; MIDDENDORF; WIGGINS, 2005).
- *Alternative Methods:* Raghavan, Albert and Kumara (2007) proposed a simple and fast detection method based on labels propagation. Initially, different labels are assigned to each vertex. At each iteration, each vertex of the network is randomly visited and each vertex receives the label with the highest occurrence among its neighbors. If a draw occurs, then the label is randomly chosen. During this process, some labels are propagated by the network and others disappear. The algorithm reaches convergence when the label of each vertex is the same label of its neighbors. Communities are formed by groups of vertices that have the same label. Oliveira *et al.* (2008) propose a divisive algorithm for community detection based on the movement of vertices. This technique maps the network topology to a dimensional space and moves the vertices in that space in such way that, at the end of the process, the vertices of the same community are positioned in close regions. Palla *et al.* (2005) propose a method to find overlapping communities based on the idea that communities tend to form clicks due to their high density of edges. On the other hand, edges that connect groups are unlikely to form clicks. Other detection techniques use information theory (ROSVALL; BERGSTROM, 2007), Bayesian inference (HOFMAN; WIGGINS, 2008), and consensus (CUPERTINO; HUERTAS; ZHAO, 2013).

Leskovec, Lang and Mahoney (2010), Orman, Labatut and Cherifi (2011) and Aldecoa and Marin (2013) present several comparisons between community detection algorithms.

Algorithms for community detection have been applied to different fields of science. Wilkinson and Huberman (2004) used community detection to find clusters of related genes. Clauset, Newman and Moore (2004) detect communities of items that are purchased together on

Amazon. Gargi *et al.* (2011) propose a method to detect communities in a network composed of millions of YouTube videos and find related videos. Traud *et al.* (2011) have applied community detection on anonymous Facebook data to investigate demographic organizations in different universities. Porter, Onnela and Mucha (2009) and Fortunato (2010) presented several other applications.

3.5 From time series to complex networks

From the characterization works of complex networks (Sec. 3.3), it was possible to study several real systems using networks. These studies allowed a better understanding of the relationship between network structure and dynamics (BOCCALETTI *et al.*, 2006; NEWMAN; BARABASI; WATTS, 2006). The interest in applying network theory to study time series has grown with the success of the application of network theory in many fields of science. In this section, we present the main methods of transforming time series into networks. Donner *et al.* (2011) divide these transformation methods into three classes: proximity networks, visibility graphs and transition networks.

- *Proximity networks*: In this approach, subsequences are represented by vertices and the connections are made using some distance function. This is one of the most used approached and it is subdivided into three categories: cyclic networks, correlation networks, and recurrence networks. We present each of these approach below.
 - *Cyclical networks*: One of the first works that proposed a mapping from time series to networks was done by Zhang and Small (2006). In this work, networks are generated from pseudo-periodic series, such as heart beats and human speech. A cycle is a subsequence between two local minimums (or maxima). For example, on electrocardiograms, each heart beat represents one cycle. The series is divided into m cycles $\{C_1, C_2, C_3, \dots, C_m\}$. Every vertex represents a cycle and the connections between vertices are performed according to the force of temporal correlation ρ between the cycles of the series. This correlation measures how similar two cycles are. Given two cycles C_i and C_j , an edge is created between them if $\rho_{ij} \geq d$, where d is a limiting parameter defined by the user. After the network construction, the authors evaluated the degree distributions $p(k)$, the average shortest paths l and clustering coefficient c . From periodic series with noise, the result of the transformation are random networks. Chaotic time series generate networks with the small-world property and are scale-free. In a second work, Zhang *et al.* (2008) used other measures such as assortativity, betweenness centrality and network motifs¹ to draw a parallel between the dynamics

¹ Motifs are connection patterns that occur in large numbers in a given real network, but do not occur so often if that same network is randomized (MILO *et al.*, 2002).

of the series and the statistical properties of the network. The authors show that it is possible to distinguish types of time series using only these measures.

- *Correlation networks*: The work of Yang and Yang (2008) consists of obtaining all the subsequences of size l from a time series. For each subsequence pair S_i and S_j , compute the correlation r_{ij} between those segments. Each segment is represented by a vertex and an edge is created between two vertices i and j if $|r_{ij}| \geq r_c$, where $r_c \in [0, 1]$ is a limiting parameter. If r_c is large, the number of connections is small and more noise is eliminated from the series. However, if r_c is too large, important edges might be removed. Thus, this work tries to find a range of values (stability region) for l and r_c in such a way that, for any value in that interval, the resulting network can reveal information of the mapped time series. After finding this interval, it is possible to notice that, for any value, the generated network presents a degree distribution that following a power law.
- *Recurrence networks*: A phase space of a time series is a mathematical space that represents all possible states of a time series. When these states are plotted versus time, the path traced by these consecutive states is called the trajectory. When a trajectory returns to a point (or area) that it has already passed through, then this time is called a recurrence. In this way, it is possible to represent all points of recurrence of a system using a recurrence graph (RP). Marwan *et al.* (2009) present a method to transform a time series into a network based on the recurrence plot. This graph is constructed from a recurrence matrix R that is binary, squared, and symmetric. R_{ij} defines whether two states $\vec{x}(i)$ and $\vec{x}(j)$ are close from each other, i.e., $R_{ij} = 1$ if $\|\vec{x}(i) - \vec{x}(j)\| \leq \epsilon$, where ϵ is an arbitrary parameter, or $R_{ij} = 0$ otherwise. In this way, the authors construct a network where each point in the phase space is represented by a vertex and the recurrences between them are represented by edges. The authors used network measures (Sec. 3.3) such as the clustering coefficient and the average shortest path to extract information and characterize network dynamics. Another approach to building networks using the recurrence matrix is to connect the nearest k neighbors (SHIMADA; KIMURA; IKEGUCHI, 2008; SMALL; ZHANG; XU, 2009). Xu, Zhang and Small (2008) Used these networks of neighbors to classify series dynamics as chaos, noise or periodic orbits.
- *Visibility graphs*: Lacasa *et al.* (2008) proposed a method to construct a graph from a time series called the visibility graph. Each observation in the series is represented by a vertex and the connections are made if a given vertex can “see” another one. The observations of the series are plotted using bar plot and an edge is created between two vertices i and j if it is possible to draw a line between the points of the series that represent them without crossing by any bar. An advantage of this method is that the network is invariant to affine transformations in the series. When applying this method to a periodic series, the resulting

network is regular. Random series generate random networks. Fractals are mapped to scale-free networks. Elsner, Jagger and Fogarty (2009) applied this mapping algorithm to transform annual series of hurricanes and detected anomalies. Shao (2010) Constructed visibility graphs from the heartbeat dynamics and used the assortativity measure (Eq. 3.5) to identify patients with heart failure. Sec. 5.1.1 presents more details about visibility graphs.

- *Transition networks*: Given a time series $X = \{x_1, \dots, x_n\}$, it can be discretized in intervals and represented by a set of symbols $L = \{L_1, \dots, L_K\}$. From these symbols, it is possible to construct a complete directed weighted graph where the vertices are the symbols from L and the weight between two symbols L_i and L_j is defined by the conditional probability $p(X_{i+1} \in L_i | X_i \in L_j)$. This network can be transformed into an unweighted graph by applying a threshold parameter $p < 1$ that defines a minimum transition probability between the symbols. Different from the other approaches, the network topology generated by this method depends only on the chosen segmentation method. Among the main works that use this approach to map series in networks are: Nicolis, Cantú and Nicolis (2005a), Dellnitz *et al.* (2006), Li and Wang (2007), Li *et al.* (2007b), and Shirazi *et al.* (2009).

3.6 Final remarks

In this chapter, we presented the concepts behind complex networks, the main network models and some measures to characterize networks. We explained what are communities in networks and the main approaches to detect them. In this thesis, the main goal is to use networks to extract information from time series. Therefore, we also presented some approaches on how to transform a time series into a network.

TIME SERIES CLUSTERING USING NETWORKS

4.1 Introduction

Clustering is an important task in time series data mining. The goal is to divide a set of time series into groups, where similar ones are put in the same cluster (ESLING; AGON, 2012). Such kind of problem has been observed in many application domains like climatology, geology, health sciences, energy consumption, failure detection, among others (LIAO, 2005).

The two desired aspects when performing time series clustering are effectiveness and efficiency (WANG *et al.*, 2013). Effectiveness can be achieved by representation methods that should be capable of dealing with high dimensional data. Efficiency is obtained by using distance functions and clustering algorithms that can accurately distinguish different time series in an efficient way. Keeping these two features in mind, many clustering algorithms have been proposed and those can be broadly classified into two approaches: data-adaptation and algorithm-adaptation (LIAO, 2005). The former extracts feature arrays from each time series and then applies a clustering algorithm in its original form. The latter uses specially designed clustering algorithms to directly handle time series. In this case, the major modification is the distance function, which should be capable of distinguishing time series.

In the original form of time series, only the local relationship among neighbor data samples can be easily identified, while long distance global relationship remains unknown in general. On the other hand, time series analysis, such as time series clustering, classification or prediction, requires not only local information but also global knowledge to capture the pattern formation of a given time series. Networks (graphs) are a powerful mechanism, which can characterize the relationship between any pair or any groups of data samples. Therefore, the transformation from time series to network representation is hopefully to present an alternative way for time series analysis. From the technical point of view, network-based clustering techniques also present

attractive advantage. Up to now, the majority of existing time series clustering techniques in literature use k -means, k -medoids or hierarchical clustering algorithms in their original forms or modified versions. The common feature of these algorithms is that they try to break data samples into clusters in such a way that the partition optimizes a criterion defined by a given distance function. As a consequence, these techniques can just find clusters of a particular shape already determined by the distance function. For example, k -means with the Euclidean distance function can only produce Gaussian distributed clusters. On the other hand, it has been shown that network-based clustering techniques can capture arbitrary cluster shapes. Network-based techniques can identify connectivity patterns of the input data and such patterns can form any shape in the Euclidean space. Finally, many community detection techniques have been proposed and some of them have even linear time complexity when the constructed network is sparse (SILVA; ZHAO, 2012). This feature also makes them attractive to time series data clustering.

In this chapter, we aim to apply network science to temporal data mining. We intend to verify the benefits of using community detection algorithms in time series data clustering. More specifically, we propose an algorithm including 4 steps of processing: (1) data normalization; (2) distance function calculation; (3) network construction, where every vertex represents a time series connected to its most similar ones using a distance function; (4) community detection, where each community represents a time series cluster. In summary, we will present in this chapter the following contributions:

- The main contribution is the proposal of using community detection in complex networks for time series clustering. For this purpose, we transform time series from time-space domain to topological domain. Networks are a general representation, which has the ability to characterize both local and global relationship among nodes (representing data samples), therefore, our approach is useful not only for time series clustering but also for other kinds of time series analysis tasks. To our knowledge, applying community detection techniques for time series clustering has not been reported in the literature;
- An extensive numerical study has been conducted in this chapter. Specifically, we study, in the time series clustering context, combinations of time series data sets, time series distance functions, network construction methods and community detection algorithms. In comparison to other time series clustering algorithms, experimental results and statistical tests show that the network-based approach presents better results.
- The proposed method presents some desired features when applied to real clustering problems. It can effectively detect shape patterns presented in time series due to the topological structure of the underlying network constructed in the clustering process. At the same time, other techniques studied in this chapter fail to identify such patterns. The proposed method is robust enough to group time series presenting similar pattern but with time shifts and/or amplitude variations.

4.2 Clustering time series with networks

The intuition behind our algorithm is simple. Each time series from a database is represented by a vertex and a distance measure is used to determine the similarity among time series and connect the most similar ones. As expected, similar time series tend to connect to each other and form communities. Thus, we can apply community detection algorithms to detect time series clusters. The idea of this algorithm is illustrated in Figure 12 and the whole process will be detailed in the following.

Figure 12 – Time series clustering using community detection in networks. First, we construct a network where every vertex represents a time series connected to its most similar time series using a distance function. Then we apply community detection algorithms to cluster time series. The communities are represented by vertices with different colors.

Source: Elaborated by the author.

More specifically, the proposed method is performed in 4 steps: (1) data normalization, (2) time series distance calculation, (3) network construction and (4) community detection. Each step is described as follows:

1. *Normalization*: The first step is a pre-processing stage that intends to scale the data set. As observed by Wang *et al.* (2013), normalization improves the search of similar time series when they have similar shapes but have different scales.
2. *Distance measures*: The second step consists of calculating the distance between each pair of time series in the data set and construct a distance matrix D , where d_{ij} is the distance between series X_i and X_j . A good choice of distance measure has a strong influence on the network construction and clustering result.
3. *Network construction*: This step intends to transform the matrix D into a network. In general, the two most used methods for network construction from a data set are the k -NN and ε -NN. The way how the network is constructed highly affects the clustering result.
4. *Community detection*: After the network is constructed, we apply community detection algorithms to search for groups of densely connected vertices to form communities. There

are plenty of community detection algorithms that use different strategies and the correct choice again affects the clustering result.

All these steps are presented in Algorithm 1.

Algorithm 1 – Time series clustering

```

1: procedure TIME_SERIES_CLUSTERING(dataset,  $k$  or  $\epsilon$ )
2:   normalization(dataset)
3:    $D \leftarrow \text{distanceMatrix}(\text{dataset})$ 
4:    $G \leftarrow \text{netConstruction}(D, k \text{ or } \epsilon)$ 
5:    $P \leftarrow \text{communityDetection}(G)$ 
6:   return  $P$ 
7: end procedure

```

The time complexity is defined as the sum of the complexities of each step of the method and it depends on the chosen algorithms and measures. Considering a data set composed by n time series all of length t , the z-score normalization of the data set can be performed in $O(nt)$. Also considering that a time series measure can be calculated in a linear time (Table 2), the network construction needs $O(n^2t)$ computations. The time complexities for the community detection algorithms (Table 3) are usually lower than quadratic and even can be linear (SILVA; ZHAO, 2012); therefore, the complexity order of the proposed method is $O(n^2t)$.

Notice that the most time-consuming process is calculating the distances between all pairs of data points, which is $O(n^2t)$. Therefore, any improvement of the nearest neighbor methods can be implemented in our method to reduce the computation time. For example, Chen, Fang and Saad (2009) propose a divide and conquer method based on Lanczos Bisection for constructing a k -NN graph with complexity bounded by $O(nt)$. Using this improvement, the complexity order of the proposed time series clustering algorithm is reduced to $O(nt)$.

4.3 Experimental evaluation

In this section, we present experimental results using the proposed methods. The experiments intend to find out the influence of the distance functions, network construction methods and community detection algorithms on time series data clustering. Finally, we compare our method to rival ones.

4.3.1 Experiment settings

For the simulations, we have used 45 time series data sets taken from the UCR repository (CHEN *et al.*, 2015). This repository is composed of real and synthetic data sets divided into training and test sets. For our experiments, we consider only the training set and we discarded the test sets. These data sets have been generated by various authors and donated to the UCR

repository. The labels of each data set are not defined by the UCR but are defined by the authors themselves according to the specific data set domain. Therefore, we have to assume that the labels are correct. Table 1 describes each data set used in our experiments.

Table 1 – The UCR time series data sets used in the experiments

Data set	Number of objects	Time series length	Number of classes
Adiac	390	176	37
Beef	30	470	5
Car	60	577	4
CBF	30	128	3
ChlorineConcentration	467	166	3
CinC_ECG_torso	40	1639	4
Coffee	28	286	2
Cricket_X	390	300	12
Cricket_Y	390	300	12
Cricket_Z	390	300	12
DiatomSizeReduction	16	345	4
ECG	100	96	2
ECGFiveDays	23	136	2
Face (all)	560	131	14
Face (four)	24	350	4
FacesUCR	200	131	14
Fish	175	463	7
Gun-Point	50	150	2
Haptics	155	1092	5
InlineSkate	100	1882	7
ItalyPowerDemand	67	24	2
Lightning-2	60	637	2
Lightning-7	70	319	7
MALLAT	55	1024	8
MedicalImages	381	99	10
MoteStrain	20	84	2
OliveOil	30	570	4
OSU Leaf	200	427	6
Plane	105	144	7
SonyAIBORobot Surface	20	70	2
SonyAIBORobot SurfaceII	27	65	2
StarLightCurves	1000	1024	3
Swedish Leaf	500	128	15
Symbols	25	398	6
Synthetic Control	300	60	6
Trace	100	275	4
Two Patterns	1000	128	4
TwoLeadECG	23	82	2
uWaveGestureLibrary_X	896	315	8
uWaveGestureLibrary_Y	896	315	8
uWaveGestureLibrary_Z	896	315	8
Wafer	1000	152	2
WordsSynonyms	267	270	25
Words 50	450	270	50
Yoga	300	426	2

Source: Chen *et al.* (2015).

The objective of the experiments was to check the performance of each combination of time series distance measures (Tab. 2), networks construction methods (ε -NN or k -NN), and community detection algorithms (Tab. 3) to each data sets. To compare the results, we use the Rand Index (RI) that measures the percentage of correct decisions made by the algorithms (HALKIDI; BATISTAKIS; VAZIRGIANNIS, 2001). The RI is defined as:

$$RI = \frac{TP + TN}{n(n-1)/2}, \quad (4.1)$$

where TP (true positive) is the numbers of pairs of time series that are correctly put in the same cluster, TN (true negative) is the number of pairs that are correctly put in different clusters and n is the size of the data set. The RI for each clustering method is calculated comparing its result to the correct clustering (labels) provided by the UCR.

We vary the parameters to find out the best clustering result, characterized by the RI index, for each data set. In the methods using k -NN, the best RI is achieved by varying parameter k from 1 to $n - 1$. In the methods using ε -NN, the best RI is achieved by varying ε from $\min(D)$ to $\max(D)$ in 100 steps of $(\max(D) - \min(D))/100$, where D is the distance matrix. For a fair comparison, the same procedure is considered in the rival methods.

The results are presented using box plots that use rectangles to represent the middle half of the data divided by the median, represented by a black horizontal line. The vertical lines represent the max and min values. Black dots inside the boxes represent the mean values. Black dots outside the boxes represent the outlier values. For comparison purpose, we use non-parametric hypothesis tests according to the methodology proposed by Demšar (2006) and we provide the p -values for the reader interpretation. In all the cases, we consider a significance level of .05, i.e., p -values $\leq .05$ indicates a strong evidence that one method statistically better (or worse) than another. On the other hand, p -values close to 1 indicates that the algorithms under comparison are statistically equivalent.

Table 2 – Time series distance functions used in the experiments

Distance	Cost	Reference
Manhattan	$O(t)$	(YI; FALOUTSOS, 2000)
Euclidean (ED)	$O(t)$	(YI; FALOUTSOS, 2000)
Infinite Norm	$O(t)$	(YI; FALOUTSOS, 2000)
Dynamic Time Warp (DTW)	$O(t^2)$	(BERNDT; CLIFFORD, 1994)
Short Time Series (STS)	$O(t)$	(MÖLLER-LEVET <i>et al.</i> , 2003b)
DISSIM	$O(t^2)$	(FRENTZOS; GRATSIAS; THEODORIDIS, 2007)
Complexity-Invariant (CID)	$O(t)$	(BATISTA <i>et al.</i> , 2014)
Wavelet Transform (DWT)	$O(t)$	(ZHANG; SMALL, 2006)
Pearson Correlation (COR)	$O(t)$	(GOLAY <i>et al.</i> , 1998)
Integrated Periodogram (INTPER)	$O(t)$	(LUCAS, 2003)

t is the length of the series;

Table 3 – Community detection algorithms used in the experiments

Algorithm	Cost	Reference
Fast Greedy	$O(n \cdot \log^2 n)$	(CLAUSET; NEWMAN; MOORE, 2004)
Multilevel	$O(m)$	(BLONDEL <i>et al.</i> , 2008)
Walktrap ^a	$O(n^2 \cdot \log n)$	(PONS; LATAPY, 2005)
Infomap ^b	$O(m)$	(ROSVALL; BERGSTROM, 2008)
Label Propagation	$O(m + n)$	(RAGHAVAN; ALBERT; KUMARA, 2007)

^a Walk length = 4;

^b Num. of trials to partition the network $nb.trials = 10$.

4.3.2 Network construction influence

The first experiment consists of evaluating the influence of the network construction on the community detection process. We verify how the parameters k and ε from the k -NN and ε -NN methods influence the network construction in order to provide a good strategy for correctly choosing these parameters and therefore get good clustering results. We start by running our method for all combinations of data sets, time series distance measures and community detection algorithms for various values of k and ε . The results are shown in Figure 13.

Figure 13 – The influence of the parameters (a) k and (b) ε on the resulting number of communities. Weak (gray) lines represent the normalized real variation of the parameter for each combination of data set, time series measure and community detection algorithm. The strongest line (blue) is an interpolation of all results, showing the average behavior. The k -NN construction method just allows discrete values of k while the ε -NN method accepts continuous values. This difference explains why the k -NN interpolation presents the sharpest decrease. In small data sets, k can assume just a few values. However, small variations of k can result in a densely connected network.

Source: Elaborated by the author.

When k and ε are small, vertices tend to make just a few connections, which, in turn, generate many network components (a component is a connected subgraph). As a result, community detection algorithms will produce a high number of clusters. On the other hand, if k and ε

are high enough, all pairs of vertices tend to be connected, leading to a fully-connected network. In this case, all vertices are considered in one big community. Examples of these behaviors are depicted in Fig. 14. So the best clustering is usually achieved when intermediate values of k and ε are chosen.

Figure 14 – An example of the influence of the network construction method on the clustering result for the coffee data set (28 time series divided in 2 classes). In this case, we constructed three k -NN networks with the INTPER measure. Vertices colors represent the communities found with the fast greedy algorithm. (a) $k = 1$ results in a disconnected network where every component is a community (RI=0.64). (b) $k = 7$ creates a connected network with 2 communities that correctly clustered all the time series (RI=1). (c) $k = 27$ creates a fully connected network where the whole network form just one big community (RI=0.48).

Source: Elaborated by the author.

We also would like to check which method is better between k -NN and ε -NN. In the following experiment, we compare the best rand index results achieved with both methods for each combination of data sets, distance measures and community detection algorithms. Tab. 4 shows some statistics of the clustering results using the two different methods. Using the Wilcoxon signed-rank test (one-tailed) (DEMŠAR, 2006), we conclude that, at a significance level of .05, the ε -NN method presents larger rand indexes (p -value $\leq .0001$), indicating that it is a better method.

Table 4 – Performance of the two network construction methods.

Network Method	Rand Index		
	Median	Mean	Std
ε -NN	0.8436	0.8133	0.1284
k -NN	0.8256	0.8012	0.1335

4.3.3 Time series distance function influence

Another factor, which may influence the performance of our method, is the time series distance function. Thus, we conduct studies to verify which one is the best for the clustering

technique presented in this section. For this purpose, we grouped the results by distance measures and plotted a boxplot. The results are shown in Figure 15.

Figure 15 – Box plot with the best rand distribution divided by measures and networks construction method. The letters in the top denote significant differences. Results that share at least one letter mean that they do not present significant differences using the Nemenyi test and a significance level of .05.

Source: Elaborated by the author.

According to the Friedman test, for both network construction methods, clustering results using different distance measures are significantly different ($p\text{-value} \leq .0001$). Hence, we proceed to the Nemenyi test to search for groups of similar measures. According to the results, DTW measure presents the best results for both network construction methods. However, we cannot statistically affirm that it is a better measure. According to the Nemenyi test, we can affirm that, at a significance level of .05, Infinite Norm, STS and INTPER present worse results than other distance measures for both methods of network construction.

4.3.4 Community detection algorithm influence

The third influence factor to our method is related to the community detection algorithm. Choosing a right algorithm can lead to better clustering results. So, we here verify which community detection algorithm is better for time series data clustering. For each combination of data sets and distance measures, we calculate the best rand index for each algorithm and plot a box plot, shown in Figure 16. The results are divided into two parts regarding the two network construction methods (k and ϵ) and, apparently, seems to be similar for both methods.

Figure 16 – Box plot with the best rand distribution divided by community detection algorithms and networks construction method. The letters in the top denote significant differences. Results that share at least one letter mean that they do not present significant differences using the Nemenyi test and a significance level of .05.

Source: Elaborated by the author.

To check whether the algorithms really have similar performance, we use the Friedman test (DEMŠAR, 2006) to compare the 5 algorithms and check whether there is a significant difference in the results. We conclude that, at a significance level of .05, for both network construction methods, the algorithms do not present similar results (p -value $\leq .0001$). Thus, the next step of our analysis consists of making a post-hoc analysis to check the difference between the algorithms. In this case, we use the Nemenyi test (DEMŠAR, 2006) to compare pairs of algorithms. For the k -NN method, we find that the Walktrap algorithm is, at a significance level of .05, better than the others. For the ϵ -NN method, the results show that the Fast Greedy and multilevel algorithms present statistically similar results and these are better than the Infomap, label propagation, and Walktrap algorithm.

4.3.5 Comparison to rival Methods

Now we present a comparison of our approach to other time series clustering methods. For this comparison, we chose the combination of network construction method, the distance function, and the community detection algorithm, which leads to the best experimental results so far. The first step consists of evaluating which algorithm achieves the best median value. We opt to compare the median instead of average because it is less sensitive to outliers (DEMŠAR, 2006). The result is presented in Tab. 5.

According to Tab. 5, the best results for the community detection approach is achieved by using the multilevel algorithm with the ϵ -NN construction method and the DTW distance

Table 5 – Performance of different combinations of networks construction methods, distance functions and community detection algorithms.

Network Method	Dist Func.	Community Detect. Alg.	Rand Index		
			Median	Mean	Std
ϵ -NN	DTW	multilevel	0.8671	0.8309	0.1309
k -NN	DTW	fastgreedy	0.8644	0.8207	0.1381
k -NN	DTW	walktrap	0.8644	0.8283	0.1297
k -NN	DTW	infomap	0.8642	0.8191	0.1431
ϵ -NN	DTW	infomap	0.8642	0.8225	0.1360
ϵ -NN	DTW	walktrap	0.8642	0.8281	0.1283
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
k -NN	L_∞	label prop.	0.8163	0.7831	0.1408
k -NN	ED	fastgreedy	0.8127	0.7958	0.1310
k -NN	STS	infomap	0.8073	0.7812	0.1415
k -NN	STS	fastgreedy	0.8016	0.7822	0.1294
k -NN	STS	multilevel	0.8016	0.7802	0.1307
k -NN	STS	label prop.	0.7980	0.7751	0.1388

Results were sorted by the median values

function. This result confirms to all the studies of influences previously presented in this thesis.

For comparison purpose, we firstly consider some classic clustering algorithms: k -medoids, complete-linkage, single-linkage, average-linkage, median-linkage, centroid-linkage and diana (GAN; MA; WU, 2007). For a fair comparison, we first find out which distance function leads to the better results for each rival method. Once again, we use the median to rank the results, which are presented in Tab. 6.

Table 6 – Best time series measures for each of the rival methods

Clustering Algorithm	Dist Func.	Rand index		
		Median	Mean	Std
Diana	DTW	0.8596	0.8167	0.1369
Centroid Linkage	DTW	0.8593	0.8075	0.1306
Single Linkage	DTW	0.8593	0.8164	0.1320
Median Linkage	DTW	0.8591	0.8075	0.1294
Average Linkage	CID	0.8575	0.8138	0.1375
k -medoids (PAM)	COR	0.8534	0.8113	0.1310
Complete Linkage	DTW	0.8501	0.8214	0.1249

Besides of those classic clustering algorithms, we also consider three up-to-date methods proposed by different authors: Zhang *et al.* (2011), Maharaj (2000) and Brandmaier (2011) (PDC). For Zhang’s method, we vary the number of clustering candidates from 1 to the size of each data set and report the best RI. In Maharaj’s method, we search for the best RI varying the significance level α from 0 to 1 in steps of 0.5. For PDC, we use the complete linkage clustering

algorithm and report the best RI from the hierarchy. Tables 7 and 8 show the best rand index for each algorithm and the corresponding data set. Figure 17 summarizes this information in a box plot.

Table 7 – Best rand index for each clustering algorithm I

	Multilevel ϵ -NN (DTW)	k -Medoids (COR)	Diana (DTW)	Complete Linkage (DTW)	Single Linkage (DTW)	Average Linkage (CID)	Median Linkage (DTW)
adiac	0.97	0.97	0.97	0.97	0.97	0.97	0.97
beef	0.83	0.83	0.83	0.83	0.83	0.83	0.83
car	0.78	0.77	0.77	0.77	0.76	0.77	0.77
cbf	0.96	0.73	0.88	0.92	0.82	0.85	0.74
chlorine_concentration	0.59	0.59	0.59	0.59	0.59	0.59	0.59
cinc_ecg_torso	0.75	0.83	0.76	0.76	0.75	0.82	0.75
coffee	0.60	0.86	0.58	0.58	0.62	0.58	0.59
cricket_x	0.92	0.92	0.92	0.92	0.92	0.92	0.92
cricket_y	0.92	0.92	0.92	0.92	0.92	0.92	0.92
cricket_z	0.92	0.92	0.92	0.92	0.92	0.92	0.92
diatom_size_reduction	0.97	0.96	0.93	0.85	0.93	1.00	0.87
ecg_five_days	0.63	0.55	0.63	0.68	0.60	0.55	0.64
ecg	0.61	0.66	0.57	0.68	0.57	0.70	0.57
face_all	0.96	0.94	0.95	0.95	0.94	0.94	0.93
face_four	0.90	0.78	0.83	0.91	0.79	0.79	0.79
faces_ucr	0.94	0.92	0.94	0.95	0.92	0.92	0.93
fish	0.87	0.87	0.87	0.87	0.87	0.88	0.87
gun	0.60	0.60	0.57	0.59	0.60	0.63	0.62
haptics	0.80	0.80	0.80	0.80	0.80	0.80	0.80
inlineskate	0.86	0.86	0.86	0.86	0.86	0.86	0.86
italy_power_demand	0.71	0.88	0.69	0.70	0.68	0.71	0.64
lighting2	0.64	0.55	0.57	0.70	0.61	0.55	0.65
lighting7	0.85	0.85	0.85	0.85	0.85	0.86	0.85
mallat	0.94	0.95	0.97	0.97	0.91	0.94	0.92
medical_images	0.69	0.69	0.69	0.69	0.69	0.69	0.69
mote_strain	0.78	0.66	0.65	0.64	0.61	0.58	0.63
oliveoil	0.88	0.91	0.86	0.82	0.87	0.90	0.87
osuleaf	0.83	0.82	0.83	0.83	0.82	0.83	0.82
plane	1.00	0.97	0.99	0.99	0.97	0.98	0.95
sony_AIBO_Robot_surface_ii	0.83	0.74	0.77	0.74	0.82	0.74	0.73
sony_AIBO_Robot_surface	0.85	0.69	0.86	0.73	0.92	0.94	0.92
starlightcurves	0.83	0.82	0.83	0.83	0.83	0.83	0.81
swedishleaf	0.94	0.94	0.94	0.94	0.94	0.94	0.94
symbols	0.97	0.95	0.97	0.97	0.97	0.96	0.97
synthetic_control	0.95	0.89	0.94	0.92	0.87	0.89	0.86
trace	0.87	0.76	0.86	0.86	0.91	0.86	0.86
two_lead_ecg	0.61	0.56	0.59	0.68	0.62	0.57	0.58
two_patterns	1.00	0.75	0.94	0.94	0.98	0.75	0.90
uwavegesturelibrary_x	0.89	0.88	0.88	0.88	0.89	0.89	0.88
uwavegesturelibrary_y	0.88	0.88	0.88	0.88	0.88	0.88	0.88
uwavegesturelibrary_z	0.88	0.88	0.88	0.88	0.89	0.89	0.88
wafer	0.82	0.82	0.82	0.82	0.82	0.82	0.83
word_synonyms	0.91	0.91	0.92	0.91	0.92	0.92	0.92
words50	0.96	0.96	0.96	0.96	0.96	0.96	0.96
yoga	0.51	0.51	0.51	0.51	0.52	0.51	0.51
Median	0.87	0.85	0.86	0.85	0.86	0.86	0.86
Mean	0.83	0.81	0.82	0.82	0.82	0.81	0.81
St.D.	0.13	0.13	0.14	0.12	0.13	0.14	0.13

We use the Wilcoxon paired test to verify if our method provides better results than the other ones. We considered the one-tailed alternative hypothesis that our method presents a higher RI. To compensate the multiple pairwise comparisons, we use the Holm-Bonferroni adjusting method (DEMŠAR, 2006). At a significance level of .05, we conclude that the community detection

Table 8 – Best rand index for each clustering algorithm II

	Centroid Linkage (DTW)	Zhang <i>et al.</i> (2011)	Maharaj (2000)	Brandmaier (2011)
adiac	0.97	0.97	0.97	0.97
beef	0.83	0.80	0.83	0.83
car	0.78	0.76	0.76	0.76
cbf	0.74	0.92	0.68	0.69
chlorine_concentration	0.59	0.59	0.59	0.59
cinc_ecg_torso	0.75	0.75	0.75	0.89
coffee	0.61	0.58	0.52	0.52
cricket_x	0.92	0.92	0.92	0.92
cricket_y	0.92	0.92	0.92	0.92
cricket_z	0.92	0.92	0.92	0.92
diatom_size_reduction	0.87	0.95	0.74	0.76
ecg_five_days	0.63	0.60	0.59	0.57
ecg	0.60	0.56	0.68	0.57
face_all	0.93	0.95	0.93	0.93
face_four	0.78	0.82	0.75	0.76
faces_ucr	0.93	0.94	0.91	0.91
fish	0.87	0.87	0.86	0.86
gun	0.58	0.59	0.51	0.54
haptics	0.80	0.79	0.80	0.80
inlineskate	0.86	0.85	0.86	0.86
italy_power_demand	0.62	0.70	0.53	0.52
lighting2	0.62	0.62	0.55	0.55
lighting7	0.85	0.86	0.84	0.84
mallat	0.91	0.94	0.87	0.89
medical_images	0.69	0.69	0.69	0.68
mote_strain	0.63	0.62	0.53	0.66
oliveoil	0.88	0.83	0.73	0.72
osuleaf	0.83	0.82	0.82	0.82
plane	0.95	1.00	0.86	0.86
sony_AIBO_Robot_surface_ii	0.73	0.79	0.71	0.51
sony_AIBO_Robot_surface	0.92	0.73	0.72	0.86
starlightcurves	0.83	0.81	0.57	0.62
swedishleaf	0.94	0.94	0.93	0.94
symbols	0.97	0.99	0.83	0.94
synthetic_control	0.86	0.94	0.84	0.84
trace	0.87	0.87	0.84	0.75
two_lead_ecg	0.58	0.58	0.52	0.55
two_patterns	0.92	0.98	0.75	0.75
uwavegesturelibrary_x	0.89	0.89	0.88	0.88
uwavegesturelibrary_y	0.88	0.88	0.88	0.88
uwavegesturelibrary_z	0.88	0.88	0.88	0.88
wafer	0.82	0.82	0.82	1.00
word_synonyms	0.92	0.91	0.91	0.91
words50	0.97	0.96	0.96	0.96
yoga	0.51	0.51	0.50	0.52
Median	0.86	0.85	0.82	0.83
Mean	0.81	0.81	0.77	0.77
St.D.	0.13	0.14	0.14	0.15

approach presents better results (p -values $\leq .02$) than k -medoids (PAM), diana, median-linkage, centroid-linkage, Zhang’s method (ZHANG *et al.*, 2011), Maharaj’s method (MAHARAJ, 2000) and PDC (BRANDMAIER, 2011). Even though our approach has presented higher median and mean values, we cannot conclude that it is statistically better than complete-linkage, single-linkage and average-linkage (p -values $\leq .32$) yet.

Figure 17 – Box plot with the comparison of different time series clustering algorithms applied to different data sets.

Source: Elaborated by the author.

4.3.6 Detecting time series clusters with time-shifts

Clustering algorithms should be capable of detecting groups of time series that have similar variations in time. To exemplify the efficiency of our method in detecting the similarity with time shifts, we consider the Cylinder-Bell-Funnel (CBF) data set, that is formed by 30 time series of length 128 divided into three groups (GEURTS, 2002). Each group is defined by a specific pattern. The cylinder group of series is characterized by a plateau, the bell group by an increasing linear ramp followed by a sharp decrease and the funnel group by a sharp increase followed by a decreasing ramp. Even composed by a small number of time series, this data set presents characteristics that make difficult the detection of similarity. In this data set, the starting time, the duration and the amplitude patterns among the time series of the same group are different. A random Gaussian noise is also added to the series to reproduce the natural behavior. Figure 18 shows the CBF data set.

Using our approach, we build a ϵ -NN ($\epsilon = 58.87$) with DTW and then apply the multi-level community detection algorithm. The result (Fig. 19) is a network with 3 communities, each one representing an original cluster of the data set. Our approach correctly finds out all the time series clusters, except the one with label “3” in Fig. 19. In this simulation, we get $RI = 0.96$ for our method. The rival method (Tab. 6) that achieves the best clustering result for this data set is the complete linkage with DTW: $RI = 0.91$.

4.3.7 Efficiency to detect shape patterns

In some cases, the similarity of time series is defined by repeating patterns that should be efficiently detected by clustering algorithms. We exemplify the efficiency of our method to

Figure 18 – The Cylinder-Bell-Funnel (CBF) data set is composed of three groups of series: (a) the cylinder group of series is characterized by a plateau, (b) the bell group by an increasing linear ramp followed by a sharp decrease and (c) the funnel group by a sharp increase followed by a decreasing ramp.

Source: Elaborated by the author.

Figure 19 – Network representation of the CBF data set using the ϵ -NN construction method ($\epsilon = 58.87$) with DTW. Vertices colors indicate the 3 communities that represent each group of time series illustrated in Fig. 18. All the 30 time series were correctly clustered, except one ($RI = 0.96$). The time series with label “3” belongs to community of color blue (bottom).

Source: Elaborated by the author.

detect different shape patterns in time series considering the two patterns data set (GEURTS, 2002). It is composed of 1000 time series of length 128 divided into four groups. These groups are characterized by the occurrence of two different patterns in a defined order: an upward step (which goes from -5 to 5) and a downward step (which goes from 5 to -5). Using these two patterns, it is possible to define four groups: UU, UD, DU, and DD. The group UU is defined by two upward steps, UD is defined by an upward step followed by a downward step, and the same logic defines DD and DU groups. According to this definition, clustering algorithms should be capable of detecting the order of patterns to correctly distinguish UD and DU. To make the problem harder, the position and duration of the patterns are randomized in such a way that there is no overlap. Around patterns, the series is characterized by an independent Gaussian noise. Figure 20 illustrates the 4 groups of the data set.

Figure 20 – The two patterns data set is composed of the sequence of two patterns: an upward (U) step, which goes from -5 to 5, and a downward (D) step, which goes from 5 to -5. The order which these patterns occur define each group: (a) UU, (b) UD, (c) DU and (d) DD.

Source: Elaborated by the author.

Using the ε -NN construction method ($\varepsilon = 44.91$) with DTW, it is possible to construct a network as shown in Fig. 21, which represents the two patterns data set. After applying the multilevel community detection algorithm to this network, we get 4 communities, representing each group of time series. All the 1000 time series are correctly clustered ($RI = 1$). The rival method (Tab. 6) that achieves the best clustering result for this data set is the single linkage with DTW ($RI = 0.97$).

Figure 21 – Network representation of the two patterns data set using the ε -NN construction method ($\varepsilon = 44.91$) with DTW. Vertices colors indicate the four communities that represent each group of time series illustrated in Fig. 20. All the 1000 time series were correctly clustered ($RI = 1$).

Source: Elaborated by the author.

4.4 Final remarks

In this chapter, we presented the benefits of using community detection algorithms to perform time series clustering. According to the experimental results, we conclude that the best results are achieved using the ε -NN construction method with the DTW distance function and the multilevel community detection algorithm among the combinations under study. We have observed that intermediate values of k and ε lead to better clustering results (Sec. 4.3.2).

For a fair comparison, we have also verified which distance function works better with each of the rival algorithms (Tab. 6). We compare those algorithms to our method using different data sets and we confirm that our method outperformed in most of the tested data sets. We have observed that our method can detect groups of series even presenting time shifts and amplitude variations. All the facts indicate that using community detection algorithms for time series data clustering is an interesting approach.

Another advantage of the proposed approach is that it can be easily fit to specific clustering problems by changing the network construction method, the distance function or the community detection algorithm. Another advantage is that general improvements on these subroutines are applicable to our method.

In this chapter, we have made statistical comparisons of clustering accuracy based on the rand index. Although it is a good measure and presents good results, it would be interesting to evaluate the simulation results using different indexes. Another point is that we have compared the best rand indexes searching from a variation of k and ε . In many real data sets, it would be infeasible to do such a searching due to the time. As future works, we plan to propose automatic

strategies for choosing the best number of neighbors (k and ϵ) and speeding up the network construction method, instead of using the naive method.

TIME SERIES CLASSIFICATION USING NETWORKS

One specific task involves the classification of time series. Many classification algorithms were proposed including the k -nearest neighbors (k -NN), decision trees, neural networks, Bayesian classifiers and SVM (XING; PEI; KEOGH, 2010). However, the simple combination of the one-nearest-neighbor (k -NN with $k = 1$) and the Dynamical Time Warping (DTW) distance function usually presents excellent results and it seems to be very difficult to beat (WANG *et al.*, 2013).

The two main aspects when dealing with time series classification are the representation methods and the similarity measures (WANG *et al.*, 2013). The former tries to reduce the amount of data and make easier the comparison of time series. The latter should efficiently capture underlying similarity of such data. Many time series representations and similarity measures were created to study time series (comparisons and references of representation methods and measures can be found in the work of Wang *et al.* (2013)), however, in our best effort, we did not find any time series classification technique based on graphs.

Graph mining is another research area intended to extract meaningful knowledge from graphs (AGGARWAL; WANG, 2010). The graph classification task is a supervised learning problem which the goal is to categorize entire graphs. Given a subset of labeled graphs in a graph dataset, the task is to build a model from the labeled graphs and use it to classify unlabeled graphs. The graph classification algorithms use two main approaches: graph kernels or feature extraction. Graph kernels measure the similarity between two graphs using random walks. Feature-based methods extract a characteristics array from each labeled graph and then use them to train a traditional classification algorithm. One of the most simple featured based classifier is the k -NN, that requires only a graph similarity measure to extract features and classify a new graph according to the most similar ones.

Using graphs to extract information from time series is a novel approach that can

capture information that traditional techniques cannot find (DONNER *et al.*, 2011). Moreover, this approach permits using the solid graph theory to extract information of sequential data. Some methods of mapping time series into networks have been recently proposed using some approaches like: correlation (ZHANG; SMALL, 2006), recurrence analysis (MARWAN *et al.*, 2009), transition probability (NICOLIS; CANTÚ; NICOLIS, 2005b), phase-space reconstruction (XU; WUNSCH, 2008), frequency transition (CAMPANHARO *et al.*, 2011) and visibility (LACASA *et al.*, 2008; LUQUE *et al.*, 2009). For a detailed review on transformation methods, we refer the interested reader to the work of (DONNER *et al.*, 2011)).

In this chapter, we aim to apply graph mining techniques to propose a new method for time series classification. More specifically, we use the visibility graph algorithm to transform a set of labeled time series into a set of labeled graphs. Then, we use the k -NN algorithm with different graph similarity measures to classify new graphs. The idea is to transform the problem of time series comparison into a graph matching problem. This transformation makes possible to use graph mining techniques and network science. Experiments show that proposed method presents good results when the time series with the same label are affine transformations from one another.

5.1 Background and related works

The increasing success of the application of graph and network theory to many scientific fields also motivated the study of time series as graphs. The main idea is to use a transformation method to convert a series from the temporal space to the topological space. With this transformation, we can use the whole graph theory and network science to extract meaningful knowledge from time series. As described in Sec. 3.5, the transformation methods can be classified in three categories: proximity networks, transition networks, and visibility graphs (DONNER *et al.*, 2011). In this chapter, we use the third class of transformation methods: the visibility graphs.

5.1.1 Visibility graphs

The natural visibility graph (NVG) algorithm was proposed by Lacasa *et al.* (2008) and consists of representing each value x_i from a series by a vertex v_i . Two vertices v_i and v_j are connected if they can “see” each other, i.e., if it is possible to draw a line between x_i and x_j that do not cross any other data height x_k :

$$\exists (v_i, v_j) \iff x_k < x_i + (x_j - x_i) \frac{k-i}{j-k} \quad \forall k \in \{i < k < j\}. \quad (5.1)$$

We illustrate an application example of the visibility graph algorithm in Figures 22. This process of transformation has some important properties:

1. *Connected*: the NVG will always be connected since two consecutive data x_i and x_{i+1} can see each other;
2. *Undirected*: the NVG algorithm creates undirected graphs by definition;
3. *Invariant under affine transformations of the series data*: the NVG is invariant under rescaling of horizontal and vertical axis and horizontal and vertical translation;
4. *Information loss*: Some information is lost during the NVG construction. For example, the series $X_1 = \{3, 1, 3, 1\}$ and $X_2 = \{3, 2, 3, 2\}$ will generate the same NVG even though they are quantitatively different.

Figure 22 – Examples of visibility graphs for the series $X = 8, 5, 5, 2, 2, 8, 1, 6, 8, 7, 1, 4, 9, 8, 4, 10, 9, 1, 10, 4$. (a) Natural Visibility Graph (NVG) (b) Horizontal Visibility Graph (HVG).

Source: Elaborated by the author.

One important point to discuss is the time complexity of transforming a time series into a visibility graph. The naive implementation consists of checking, for every pair of vertices, if all the vertices between them do not cross the line that connects these two vertices, resulting in a complexity $O(n^3)$. This complexity is impractical for long time series. However, some advances in computational geometry made possible to speed up the construction of VGs. One of the interesting nontrivial solutions was proposed by Lee (1978) that construct the VG in time $O(n^2 \log n)$. For a complete review on visibility algorithms, the interested reader may check (GHOSH, 2007).

5.1.2 Graph similarity

After presenting some methods to transform time series into a graph, now we present some measures to compute the similarity between two graphs. This comparison is essential in graph mining. In fact, instead of computing similarity, we believe that computing the distance between two graphs is more straightforward. These distance functions can be divided into three classes: isomorphism, inexact matching, and topological descriptors (BORGWARDT, 2007).

The first class of measures is based on graph isomorphism.

Definition 38 (Graph Isomorphism). Two graphs $G = (V, E)$ and $G' = (V', E')$ are isomorphic if there exists a bijection $f : V \rightarrow V'$ with $(v_i, v_j) \in E \Leftrightarrow (f(v_i), f(v_j)) \in E' \forall v_i, v_j \in V$.

This topological comparison can be used as a simple distance function where $d(G, G') = 0$ if the graphs G and G' are isomorphic (match) or 1 otherwise (do not match). This binary measure is known as exact graph matching (CONTE *et al.*, 2004). One problem with this approach is the time complexity required for verifying the isomorphism since no polynomial solution is known. Instead of comparing the whole graph, some related measures compare subgraphs. However, graph isomorphism is not commonly used in practical because graphs in real-world often have noise and rarely present an exact matching (CONTE *et al.*, 2004).

The second class of measures tries to solve the problem of the noise by considering a tolerant error. The inexact matching measures the difference between two graphs using an edit distance, defined as the cost of the least expensive sequence of edit operations that are needed to transform one graph into another. The idea is to count the minimum number of node and edges insertions, deletions and relabelings to transform one graph into the other (GAO *et al.*, 2010b). The Jaccard index is one of most well know measures. This measure computes the number of vertices and edges overlaps between two graphs (PAPADIMITRIOU; DASDAN; GARCIA-MOLINA, 2010). Another related measure is the Gromov-Hausdorff distance (GHD) (MEMOLI, 2008). This measure compares how far two compact metric spaces are from being isometric. It is commonly used to compare different shapes. Considering the graph structure as a metric space, the GHD can be applied to weighted graphs comparison. Given two finite metric spaces ρ_1 and ρ_2 , the l^p Gromov-Hausdorff distance is defined as:

$$D_p(\rho_1, \rho_2) = \inf\{\|(|d(\phi_1(x), \phi_2(x))|)_{x \in X}\|_p\}, \quad (5.2)$$

where the infimum is taken over all isometric embeddings (preserve the same distance) ϕ_1 of (X, ρ_1) , ϕ_2 of (X, ρ_2) into a common metric space (Y, d) (LIEBSCHER, 2015; MEMOLI, 2007).

The third class of measures consists of extracting feature vectors that describe the topological properties of the graphs. This vectors are called topological descriptors and can be used to compare graphs (SOUNDARAJAN; ELIASSI-RAD; GALLAGHER, 2014). In general,

what these measures do is to extract two n -dimensional feature vectors p and q from graphs G_1 and G_2 respectively and compare them using, for example, the Canberra distance. Different measures use different descriptors to compute distance. Features are extracted observing local characteristics, like vertex degree and vertex transitivity, to global features like community structures, average shortest path, average degree, etc. For a survey on network measures, we refer the interested reader to (COSTA; RODRIGUES, 2007).

Another classification is related to the vertex correspondence. Some distance functions require that the vertices and edges of the graphs should have unique labels before the comparison. These functions just measure the correspondent difference between the label vertices and edges. The advantage is that they do not require to discover first which is the best matching between the two graphs and therefore are faster. The disadvantage is that they cannot detect similarity when graphs are similar (inexact matching) but have different labels.

5.1.3 Graph classification

The problem of graph classification is a supervised learning problem which the goal is to categorize entire graphs. In this task, there is a labeled subset in a graph dataset that is used to build a model to predict the label of a whole uncategorized graph. We formally define the problem as:

Definition 39 (Graph Classification). Given a set of N training examples $\mathbb{G} = \{(G_1, y_1), \dots, (G_N, y_N)\}$ where G_i is a graph associated to a label y_i from a set of p labels $Y \in \{1, 2, \dots, p\}$, the graph classification problem consists of inducing a mapping function $f(G) : \mathbb{G} \rightarrow Y$ that correctly map a graph to its label. This function is used to assign labels to an unlabeled graphs.

The graph classification algorithms can be broadly classified into featured-based and distance-based. The former extract feature vector from the graph training set and use traditional classification algorithms.

The distance-based algorithms use distances functions to measure and evaluate the similarity of pairs of graphs. Examples of algorithms are the k -nearest neighbors, graph kernels, and graph embedding. The k -NN classifier is one the most simple supervised algorithms because it only needs a distance function to classify objects. Therefore, it is very convenient for graphs classification. Given a labeled set of training graphs, the k -NN classifier assigns an unknown graph to the class that occurs most frequently among the k graphs from the training set that have the smallest distances to the unknown graph. Due to its simplicity, we chose to use our approach.

5.1.4 Time series classification

The time series classification problem is similar to the graph classification problem. The idea is to train a classifier using a previously labeled set and use this classifier to label new

series. The time series classification methods can be broadly classified into three categories: featured-based, distance-based and model-based algorithms (XING; PEI; KEOGH, 2010).

Featured-based methods extract feature arrays from time series and use traditional classification algorithms, like decision trees and neural networks. The difficulty of these methods is to correctly obtain the descriptive features that distinguish each class. One example of a feature are the Shapelets, which are subsequences which can maximally represent a class (YE; KEOGH, 2009).

Distance-based methods define a distance function to measure the dissimilarity between pairs of time series. Examples include k -NN and support vector machines (SVM). These methods are widely used because they just require a time series distance function. The two mostly used time series distance functions are the Euclidean and Dynamic Time Warping (DTW) distances (WANG *et al.*, 2013). Although its simplicity, the 1-NN (k -NN with $k = 1$) algorithm with DTW provides better results than classifiers such as neural networks and SVM.

Model-based methods assume that all time series of the same class were generated by an underlying model. Each model describes the probability distribution of the process generating the data from each class. In the training step, the algorithms learn which parameters describe the probability distribution. In the classification step, an unlabeled series is assigned to the class with the highest probability. Examples include the Naive Bayes, Markov models and Hidden Markov Models (HMM).

5.2 Method description

The idea of our method is to first transform a set of labeled time series $\mathbb{X} = \{(X_1, y_1), \dots, (X_N, y_N)\}$ into a set of labeled graphs $\mathbb{G} = \{(G_1, y_1), \dots, (G_N, y_N)\}$. In order to classify a new time series X , we use a graph similarity measure to search for the k most similar (k -NN) graphs in training set $\mathbb{G}$. The label with more occurrences in the (k -NN) graphs is assigned to the new graph. We illustrate his strategy in Fig. 23.

By definition, the visibility algorithm creates unlabeled graphs. As seen in Section 5.1.2, the problem of unlabeled graphs comparison is more difficult then comparing labeled graphs. Thus, we can simplify this problem by using the natural order of the series to label vertices and edges in order to make the graphs comparison easier. The vertices labels $L_V = \{1, 2, \dots, n\}$ are simply defined as the order index of observations in the series. This simply labeling process permits to use graph distance functions that require vertex correspondence.

We also considered adding weights to the edges in order to add more information about the relying time series. Every edge has a weight w_{ij} that is the Euclidean distance between the

Figure 23 – Time series classification using networks. The idea is to transform labeled training set of time series into a set of networks (different colors represent different labels). Then, an unlabeled time series is transformed into a network and calculated its similarity to the other networks from the training data set. Finally, the new time series can be classified using the k -NN algorithm.

two data points represented by the vertices v_i and v_j :

$$w_{ij} = \sqrt{(t_j - t_i)^2 + (x_j - x_i)^2}. \quad (5.3)$$

After constructing all networks, the classification is performed by similarity. In this chapter, we used the l^p Gromov-Hausdorff distance described in Eq. 5.2. In our tests, we used the Euclidean type metric (l^2 GHD). One motivation for using this distance function is the capability of comparing weighted graphs. The second motivation is the capacity of finding similarity in unlabeled graphs (isomorphism). We wanted to use a distance function that works similar to the DTW. This measure should first match the vertices before the comparison. Both GHD and DTW distances try first to match graphs and time series respectively.

5.3 Experimental evaluation

5.3.1 Toy case

We started the experiments showing a toy case where 1-NN with DTW fails to classify. Consider the data set illustrated in the top part of Figure 24. It is composed of four time series with two different shapes. The blue and the red ones are trended sinusoidal and cosinusoidal curves respectively. The goal here is to correctly classify the two detrended time series in the bottom part of Figure 24. The k -NN classifier with DTW algorithm misclassifies them.

In this case, when the series are transformed to the topological space, using the NVG algorithm, we can clearly see the similarity between the test cases and training ones. Figure 25 illustrates the respective NVGs for the time series in Fig. 24. Using the 1-NN with GHD function, we can successfully classify them. Our method works here because time series from

Figure 24 – A case where 1-NN with DTW fails to classify time series. The goal is to classify the times series 5 and 6 according to its shape similarity to class blue: sin (series 1) or red (2). The 1-NN with DTW algorithm misclassifies it as belonging to class red (2).

different classes are affine transformations from each other. It is clear here that a simple detrended preprocessing would make DTW work for this case, but there are more complicated situations that it would not work.

Figure 25 – Two NVGs constructed using the 2 time series in test dataset in Fig. 24 (numbers 5 and 6 respectively). (a) NGV for all sinusoidal time series in red (1, 3 and 5), (b) NGV for all cos-sinusoidal time series in blue (2, 4 and 6). Using the 1-NN we can correctly classify this time series.

Source: Elaborated by the author.

5.3.2 Real case

Here we present the results for a real dataset where our approach presents better results than DTW and 1-NN classifier. We used a dataset composed of time series extracted from binary images of beetles and flies (CHEN *et al.*, 2015). Each image was transformed into a time series by getting the distance from the outline and the center of the image. Each time series is composed of 512 values and were normalized. The dataset is composed by 20 training instances and 20 test instances illustrated in Fig.26. The problem consists of using this dataset to detect if an unknown outline is a beetle or fly.

Figure 26 – Time series dataset (a) composed by distances between the outline and the image center of beetles (b) and (c) flies.

Source: Adapted from Bober (2001).

After running 1-NN with both DTW and GHD in this dataset, our approach provides better results. The accuracy results for DTW and GHD are 70% and 75% respectively. The dataset explains this result. Some images are transformations (rotation) from one another, what leads to similar visibility graphs. In fig. 27 we show a specific case where DTW fails to classify a time series from the BeetleFly dataset. The goal is to classify the test instance number 12 (red). After transforming these three time series to VGs it is possible to compare them using GHD. DTW considers the training instance number 7 (green) as the most similar while GHD considers the training instance number 11, which provides the correct label.

5.4 Final remarks

In this chapter, we presented a new time series classification algorithm based on networks. We adapted the VG by adding weights that correspond to the distances between two values in the series. These weights add more information about the series to the resulting VG. In order to compare two graphs, we need a graph distance function that take into account the weights. We also wanted to use a graph distance function that takes into account some inexact matching. The idea was to transform the time series alignment problem (done by DTW) into a graph matching problem. The classification process was also done by similarity using k -NN. The advantage of this transformation is that makes possible the use of graph similarity measures and the whole graph theory framework to extract information from temporal data.

Our results suggest that our method presents good results when the time series from the same label (class) are affine transformations from one another. In these cases, normalization or trend removal are not required.

The results here presented are still initial, but sufficient to show that is possible to use networks to perform time series classification. The results here presented opens opportunity of research in many directions. Some ideas include the use different methods to map time series into graphs. Testing other graph distance functions. Applying dimension reduction in graphs before comparing them. Using different graph classification algorithms like graph kernels and graph-embedding.

Figure 27 – A case where 1-NN and DTW fail to classify a time series from the BeetleFly dataset. The objective is to classify the time series in red (test 12). Networks and time series with the same color indicate the correspondent transformation. On the left of each network, there is a square that represents the normalized adjacency matrix of the network. The darker the color of the edge, the stronger the connection. DTW considers the time series in green (train 7) as the most similar. Using GHD, the distances between the test instance (red) and the two training instances 7 and 11 are 33.60 and 26.22 respectively. In this case, GHD correctly assigns the label according to similarity.

Source: Elaborated by the author.

PERIODICITY DETECTION USING NETWORKS

6.1 Introduction

Complex networks are a scientific field motivated by observations in real networks. Numerous systems have been represented with vertices and edges then studied in order to search for common properties. One particular study involves the representation of time series as networks. The main idea is to transform a time series into a network and explore its properties using complex networks techniques. Traditional techniques of time series analysis make a summarization of the time series characteristics and use them to provide information about the dynamics in the series. Using networks to study time series is a novel approach that can capture information that traditional techniques cannot find (DONNER *et al.*, 2011).

Transforming series into networks is however just the first step to study time series with networks. The next step consists of using complex network theory to make time series analysis. One typical problem studied in time series analysis is the periodicity detection. A period is a repeating pattern in the series. This pattern is extensively observed in many real sequential data from different research areas like oceanography, astronomy, meteorology, medicine and engineering (VLACHOS; YU; CASTELLI, 2005). Many techniques of periodicity detection have been proposed, however, the majority of these methods fail to detect periods in noisy data (PARTHASARATHY; MEHTA; SRINIVASAN, 2006).

In this chapter, we propose a method for periodicity detection based on community detection in networks. This method consists of dividing a time series into intervals that are represented by vertices. Two vertices are connected if there are two consecutive observations in the time series positioned in two different intervals. Once the network is created, community detection can be applied in order to cluster highly connected ranges. Missing and noisy data are handled by the discretization process and the community detection process. We applied

this method to a temperature time series and observed that is possible to use complex networks techniques to perform periodicity detection. In summary, in this chapter we present the following contributions:

- A new form of transformation from a time series to network based on intervals;
- A new algorithm of periodicity detection in series based on community detection in networks.

The remainder of this chapter is organized as follows. First, we present in section 6.2 some related works. In section 6.3, we introduce the new form of transformation from a time series into a network and how to use it in order to detect periodicity. In section 6.4, we show the application of this method to a sinusoidal series as example. We also show in this section the results from the application of the method to a real world time series. Finally, we enumerate some final remarks in section 6.5.

6.2 Related works

Time series periodicity detection is an old problem that motivates many researchers from different areas, including data mining researchers. Vlachos *et al.* (2006) present an algorithm, called AUTOPERIOD, that performs the detection in two steps. First, the algorithm uses a periodogram to elect candidate periods. The second phase consists of using autocorrelation to verify if the candidate periods are valid periods. Papadimitriou, Brockwell and Faloutsos (2003) present an algorithm called AWSO to detect periodicity using wavelets. Rasheed, Alshalalfa and Alhajj (2011) use suffix trees to detect periodicity in complex time $O(kn^2)$, where k is the maximum length of periodic pattern.

Many techniques (HAN; GONG; YIN, 1998; ELFEKY; AREF; ELMAGARMID, 2005a; ELFEKY; AREF; ELMAGARMID, 2005b) turn the real values of the series into discrete values. Considering an alphabet Σ , the discretization process consists of mapping distinct ranges of values of a time series X to symbols from Σ . For example, considering three ranges $a = [0, 3)$, $b = [3, 6)$, $c = [6, 9]$ and the alphabet $\Sigma = \{a, b, c\}$, the series $X = \{1, 3, 8, 4, 2\}$ can be discretized to $X = \{a, b, c, b, a\}$. Elfeky, Aref and Elmagarmid (2005a) presented an algorithm for periodicity detection based on convolution. The idea of these algorithms is to shift and compare time series. For example, the discrete series $X = \{a, b, c, \mathbf{a}, \mathbf{b}, d, \mathbf{a}, \mathbf{b}, c\}$ can be shifted three positions resulting in $X' = \{*, *, *, \mathbf{a}, \mathbf{b}, c, \mathbf{a}, \mathbf{b}, d\}$. Comparing X to X' , there will be four matches out of six possible matches. Therefore, X' is considered a candidate period. This algorithm can be performed in time $O(n \log n)$, where n is the length of the series.

The algorithm proposed by Nuñez *et al.* (2012) is the most similar to our approach. First, the authors transform the series into a network called horizontal visibility graph (LUQUE *et*

al., 2009). Every observation of the series is represented by a vertex and a connection is made between two vertices if they can “see” each other. In other words, the observations of the series are plotted using a bar chart and a connection is made if it is possible to draw a horizontal line between two observations that do not cross any bar. Once the network was constructed, the authors use some properties of the resulting graph to build a noise reduction filter and detect periodicity.

6.3 Method description

In this Section, we present a method for detection of periodicity in time series based on networks. As many techniques, we also make the discretization of the series. However, instead of using an alphabet to represent the ranges of the series, we use a sequence of integers $\{1, \dots, n\}$. After the discretization process, we use these integer labels to build an adjacency matrix and map every range of the series to a vertex of a network. The network topology is built according to the change of intervals in the series. Once the network was constructed, we use a community detection algorithm to cluster high connected ranges. These communities indicate frequent changes in the ranges and can be used to detect periodicity, even in the presence of noise. The whole process will be detailed as follows.

The first part of the algorithm consists of transforming a time series X into a network. This process consists of dividing X in n distinct ranges with the same length. There are many strategies to automatically finding the number of ranges in a dataset (GARCIA *et al.*, 2013). We chose one of the most simple ways of equal-length binning: the square root of the dataset length. With this simple method is possible to divide a dataset in $n = \lceil \sqrt{t} \rceil$ ranges with the same length. These ranges can be represented by a set $R = \{r_1, \dots, r_n\}$ ordered from low to high ranges. Every range $r_i \in R$ will be represented by a vertex $v_i \in V$ in the network $G(V, E)$. For this transformation, we may label every range of R as a sequence of integers $\{1, \dots, n\}$ in the same order they appeared in R . These labels will be used to represent the indexes of an adjacency matrix A . Initially, A is considered a null matrix. A connection between two vertices v_q and v_w is made if there are two consecutive observations x_i and x_{i+1} in the series located in the different ranges r_q and r_w . In this case $A_{qw} = A_{wq} = 1$. Repeating this process to all pairs of consecutive observations in X , we have the resulting network. The network generation procedure is described in the Algorithm 2.

Once the network was generated, we may apply community detection algorithms in order to cluster highly connected intervals. The result of this process will lead to a set of communities P that will be used to detect periods. For the sake of simplicity, we use the term “community of an element x_i ” to refer to the community which the vertex that represents the range where x_i is inserted. The process of detection starts by getting the community of the first element x_1 of the series and mark this community as visited. Then we get the community of the second element x_2

Algorithm 2 – Network generation.

```

1: procedure NETWORK_GENERATION( $X$ )
2:    $n \leftarrow \lceil \sqrt{t} \rceil$ 
3:    $R \leftarrow \text{ranges}(X, n)$ 
4:   for  $i \leftarrow 2$  to  $t$  do
5:      $r_1 \leftarrow \text{range}(R, x_{i-1})$ 
6:      $r_2 \leftarrow \text{range}(R, x_i)$ 
7:     if  $r_1 \neq r_2$  then
8:        $v_1 \leftarrow \text{label}(r_1)$ 
9:        $v_2 \leftarrow \text{label}(r_2)$ 
10:       $A_{v_1 v_2} \leftarrow 1$ 
11:       $A_{v_2 v_1} \leftarrow 1$ 
12:    end if
13:  end for
14:  return  $A$ 
15: end procedure

```

and also mark this community as visited. This iterative process goes through the entire series X and a period is considered every subsequence S^X that starts in a community, visits all the other communities and comes back to the first one. Using this algorithm, it is possible to break the series in a set of periods P . This detecting periodicity procedure is described in the Algorithm 3.

Algorithm 3 – Periodicity detection.

```

1: procedure PERIODICITY_DETECTION( $X, A$ )
2:    $C \leftarrow \text{communityDetection}(A)$ 
3:    $C_{\text{visited}} \leftarrow \emptyset$ 
4:    $P \leftarrow \emptyset$ 
5:    $C_{\text{first}} \leftarrow \text{community}(C, x_1)$ 
6:   for  $i \leftarrow 2$  to  $t$  do
7:      $C_{\text{current}} \leftarrow \text{community}(C, x_i)$ 
8:      $C_{\text{visited}} \leftarrow C_{\text{visited}} \cup C_{\text{current}}$ 
9:     if  $C_{\text{current}} = C_{\text{first}}$  and  $|C_{\text{visited}}| = |C|$  then
10:       $P \leftarrow P \cup \{i\}$ 
11:       $C_{\text{first}} \leftarrow \text{community}(C, x_i)$ 
12:       $C_{\text{visited}} \leftarrow \emptyset$ 
13:    end if
14:  end for
15:  return  $P$ 
16: end procedure

```

An important aspect to discuss is how the algorithm deals with noisy data. In real world time series, two common problems occur: missing and imprecise data. These problems are very frequent due to the inefficient measurement techniques and storage problems. In our algorithm, these problems are handled by the discretization and community detection processes. Small variations in the data are solved by the discretization process that transforms all data in the same range into the same vertex. The discretization also addresses the issue of missing data in the case

where the previous and posterior elements are located in the same range. For example, if there is a repeating pattern $S = \{1, 2, 3\}$ and all the elements 1, 2 and 3 are located in the same range, if the value 2 is missing in same subsequence of the series ($\{1, 3\}$), the result will be the same, since the algorithm just considers ranges variations. The missing data is also handled by the community detection process. There are some specific cases that the result of the community detection is the same for different networks. For example, the repeating pattern $S = \{1, 2, 3, 4, 5, 6, 5, 4, 3, 1\}$ will generate the network depicted in Fig 28a. If the value 5 is missing in some subsequence, the resulting network will be the one shown in Fig. 28b. In this cases, the result of periodicity detection will also be the same.

Figure 28 – Result for the transformation process of two different time series into networks. In both cases, the result of community detection (identified by colors) is the same. This feature is used to handle missing data in the series.

Source: Elaborated by the author.

Figure 29 – Periodicity detection in a sinusoidal dataset. (a) The dataset was generated using the function $\sin(x)$ from 0 to 6π sampled at intervals of 0.10. and divided in $n = 14$ ranges. (b) Network resulted of the transformation from the sinusoidal dataset. Different colors represent the 4 different communities found with the fast greedy algorithm (CLAUSET; NEWMAN; MOORE, 2004). (c) Result of the periodicity detection algorithm. Vertical red lines represent the breaks in the periods found by the method. Every period $P = \{64, 127\}$ in the sinusoidal series was successfully found with the algorithm.

Source: Elaborated by the author.

The time complexity of the algorithm is linear if the community detection process is not considered. The network generation procedure makes $n - 1$ comparisons in the series data to

create the edges. The periodicity detection procedure also makes $n - 1$ comparisons to search for changes in communities. The result is a total complexity of $O(n)$. For the experiments, we used the fast greedy algorithm (CLAUSET; NEWMAN; MOORE, 2004) that detects communities in time $O(m.d.\log n)$, where the d is depth of the dendrogram that describes the network's community structure. Considering the community detection process, the time complexity is $O(\max(m.d.\log n, n)) = O(m.d.\log n)$.

6.4 Experimental evaluation

In this Section, we present the application of the proposed method of periodicity detection for two different series: an artificial sinusoidal and a real historical temperature dataset.

6.4.1 Sinusoidal series

Initially we consider an artificial sampled sinusoidal series as an example of the method application. The dataset was generated using the $\sin(x)$ function sampled from $x = 0$ to $6x = \pi$ in intervals of 0.1. It resulted in a dataset composed by 189 elements where every period has length of 63. In Fig. 29a, we exhibit the dataset, where the x axis represents the index of the element and y axis represents the real value of the series. This dataset was divided in $n = 14$ ranges labeled from 1 to 14 where interval 1 is the most low one that used by Alg. 2 to transform the series into to a network ($n = 14$), illustrated in Fig. 29b. Every vertex color represents the community whose vertex is inserted. In this experiment, we used the fast greedy algorithm for community detection (CLAUSET; NEWMAN; MOORE, 2004) that automatically find the best number of communities. In this case, 4 communities were found. Once the network was constructed, we can use Alg. 3 to find periods that are depicted in Fig. 29c by red lines that divide the series. The algorithm correctly detected every period of the series $P = \{64, 127\}$ that correspond respectively to the periods 2π and 4π of function $\sin(x)$.

6.4.2 Temperature series

The second application consisted of applying the proposed method to a real dataset composed by the historical temperatures in the city of São Carlos - SP, Brazil (INMET - Instituto Nacional de Meteorologia, 2013). This dataset is composed by the average temperatures from month between 1983 and 2013. This dataset has some missing data and it was interesting to test if the method is sensitive to this usual problem. Specifically, the years 1985, 1992, 1993, 1998 and 2000 have just 0, 8, 10, 11 and 10 values respectively. All the other years have 12 values resulting in a dataset with a total size of 351 values. This dataset is exhibited in Fig. 30a, divided in $n = 19$ ranges. Using Algorithm 2, is possible to transform the dataset to a network (Fig. 30b) and then apply community detection. In this case, two communities were found, separating the coldest from the hottest temperatures. Once the network was created and the communities

were found, is possible to apply the Algorithm 3 to detect the periods, that are illustrated in Fig. 30c. The dataset is composed by 90% of periods with length 12 and our algorithm detected approximately 70% of periods with length 12.

6.5 Final remarks

In this chapter, we presented an algorithm for periodicity detection based on community detection. The main difference for this technique to the traditional methods is that it is based on network theory. The algorithm performs the detection in two steps. First, the a times series is transformed into a network. Then, communities are detected and used to locate the periodicity. An advantage of this method is that it is parameter free and do not require a priori knowledge of the dataset to tune the algorithm. This method was empirically evaluated with two dataset and presented good results, even in the presence of noisy data.

As future works some points are suggested. It would be interesting to use network metrics to study the network generated by the transformation algorithm in order to search for patterns between series and networks. For the same purpose, it would be interesting to apply this method to other types of series and compare the results for different periodicity detection algorithms. In short, the network representation leads to a new approach to time series analysis that may be explored.

Figure 30 – Application of the proposed method to detect periodicity in a real dataset composed by the average monthly temperatures of the city of São Carlos - SP, Brazil. Figure (a) shows the plot of the series divided in $n = 19$ intervals separated with horizontal dashed lines. These intervals are labeled from 1 to 19 where interval 1 is the most low one. Vertices colors represent each community in the networks. (b) Exhibited the network generated from the temperature time series. Colors represent different communities. (c) Shows the result of periodicity detection where every period is separated with red vertical lines.

(a)

(b)

(c)

Source: Elaborated by the author.

CONCLUSION

The central question behind this thesis is: “*Can complex networks bring advantages to time series mining?*”. The answer is positive. In this thesis, we divided this broad question into tasks related to each time series data mining. We made the same question for each task and presented the results for 3 of them: clustering, classification and periodicity detection. One motivation behind this study came from the fact that networks can represent complex systems in a unified way. This form of representation made possible understanding complex behaviors in many different systems that previous methods were not able to detect. The second motivation comes from the great potential of networks as a machine learning tool. Networks have been widely used in clustering and classification tasks. As we observed in this thesis, networks can also be used to perform time series data mining.

In the context of time series clustering (Sec. 4), we tested many different combinations of network construction methods, similarity measures, and community detection algorithms in several synthetic and real dataset. We observed that the best combination of methods is the ε -NN construction method with the DTW distance function and the multilevel community detection algorithm. This approach presented results statistically better than traditional clustering methods. This method can be easily adapted for specific clustering problems by changing the network construction method, the time series distance function or the community detection algorithm. The conclusion is that the network structure and community detection algorithms can better detect time series clusters. These results are observed due to the capability of community detection algorithms to find groups of different shapes and sizes.

We also tested methods for times series classification (Sec. 5). For the classification task, we transformed ever time series from a database into a set of visibility graphs. We wanted to transform the problem of time series classification into a graph classification problem. With this approach, we could use the benefits of the transformation method, graph theory, network science, and network mining techniques. As discussed in the literature, the 1-NN classifier with DTW distance function is very difficult to beat. In this thesis, we showed that using the visibility graph

and the GHD is possible to get better results in some situations. We used the GHD because we wanted a distance function that considered graph matching before comparing them. The graph matching problem here is analogous to the time series matching performed DTW. When time series from the same class (label) are affine transformations from one another, the resulting visibility graph is similar, leading to good classification results.

In section 5, we presented the results for periodicity detection in time series using networks. We used a similar approach applied to the time series classification task. First of all, time series were transformed to time visibility graphs. One feature of the VG is that the local maxima in the series are usually mapped to highly connected vertices that link two communities. We used this characteristic to propose a periodicity detection method base on community detection algorithms. Communities represent the periods of the time series. This method can detect periods in noisy data and does not require parameters.

The methods and results presented in this thesis show that it is possible to transform time series into networks and use network science to perform time series data mining. In some cases, this approach can provide better results than traditional methods. Considering the number of tasks and problems related to time series data mining, the approach used in this thesis can be applied to many other problems. In the following section (7.1) we suggest a list of possible extensions to this work.

7.1 Future works

- *Summarization*: Techniques of summarization (or segmentation) of series consist of providing a generalized information of the characteristics of the series. The goal is to find data that does not matter so much to the series and remove them to reduce the number of data in a time series and simplify its analysis. These techniques can be used independently in data pre-processing steps in data mining or as ancillary techniques for other tasks.
- *Anomaly detection*: Outliers Are data from a series that differ greatly from the standard. These points can be generated by measurement errors, in which case they can be discarded in the pre-processing stage. Or, these anomalies may represent points of interest in a given series. Considering, for example, the number of credit card purchases, a high value that differs from the purchase pattern may indicate the theft and misuse of the card. In this way, the detection of outliers can be used as a tool for the data mining in time series. In this step, we intend to develop techniques to find anomalous series in a database and techniques to find anomalous subsequences in one or more series.
- *Search*: This type of problem is related to the retrieval of information in series. A growing number of studies have been carried out in this area (DING *et al.*, 2008; LIAN; CHEN, 2008; SHIEH; KEOGH, 2008). This task consists of finding specific data in a time series or finding a series in a set of series. In both cases the search is done using a query that

represents the desired information. To perform the search, it is necessary to organize (index) the data of a series in a strategic way that facilitates the task. Once the database has been indexed, it becomes necessary to create search algorithms. Thus, you can create series indexing methods across complex networks and use those networks to search for content.

- *Association rules*: It seeks to identify cause and effect relationships between the data. In time series, this problem consists of finding patterns in one or more series. When only one series is considered, the goal is to identify relationships between subsequences. For example, in stock market analysis, various patterns of price variation have been cataloged and used for forecasting (EDWARDS; MAGEE; BASSETTI, 2007). That is, probabilistic mappings of the type are made: given a subsequence S_1 so exists $x\%$ of chance to occur S_2 in the following. In multivariate series, the correlations between variables can be represented by association rules. For example, considering that the price of a stock A_1 is increasing and that A_2 is decreasing, then there is $x\%$ of the stock price A_3 also decreasing. Several works have been developed in this area (CHEN; PETROUNIAS, 2000; LEE; LIN; CHEN, 2001; SARKER *et al.*, 2003; MÖRCHEN; ULTSCH, 2007).
- *Multivariate time series clustering*: In this thesis we presented the results for univariate time series clustering. This approach can be easily extended to multivariate time series in the following ways: (1) changing the time series distance function. In this case, we just need to use a new distance function designed for multivariate time series. The network construction method and the clustering method remain the same. (2) Changing the clustering method. In this case, a new clustering method has to be developed to deal with every series variables. One possible way is to apply our method to each variable and then use some criteria to merge the clustering results.
- *Data streams*: When the dynamics of the series are called into question, the analysis becomes even more difficult. In many real cases, the series does not have an end value, but they are infinite data streams that must be analyzed. These data usually come in large quantities. In this case, limiting in-memory storage makes analysis more difficult. Whether it is to group, classify or any other task of data mining in time series, in this type of problem it becomes necessary to use some strategy of discarding or summarizing data. Several reviews have been published in this area (HULTEN; SPENCER; DOMINGOS, 2001; GABER; ZASLAVSKY; KRISHNASWAMY, 2005; AGGARWAL, 2007; GAMA, 2010; KHAN, 2010).

7.2 Publications

Here we present all the peer-reviewed publications created during the doctorate.

1. FERREIRA, L. N.; ZHAO, L. Time series clustering via community detection in networks. **Information Sciences**, v. 326, p. 227 – 242, 2016. ISSN 0020-0255.
2. FERREIRA, L. N.; ZHAO, L. A time series clustering technique based on community detection in networks. *Procedia Computer Science* (**Proceedings of International Neural Network Society - Conference on Big Data (INNS-BigData)**), v. 53, p. 183 – 190, 2015. ISSN 1877-0509.
3. FERREIRA, L. N.; ZHAO, L. Time Series Clustering via Community Detection in Complex Networks. In: **Proceedings of International School and Conference on Network Science (NetSci-X 2015)**. Rio de Janeiro, RJ, Brazil, 2015
4. FERREIRA, L. N.; ZHAO, L. Time Series Classification with Complex Networks. In: **Proceedings of International School and Conference on Network Science (NetSci-X 2015)**. Rio de Janeiro, RJ, Brazil, 2015
5. FERREIRA, L. N.; ZHAO, L. Detecting time series periodicity using complex networks. In: **2014 Brazilian Conference on Intelligent Systems**, 2014. p. 402–407.
6. ALVES, L. G. A. et al. Long-range correlations and fractal dynamics in *c. elegans*: Changes with aging and stress. **Phys. Rev. E**, American Physical Society, v. 96, p. 022417, Aug 2017.

BIBLIOGRAPHY

ADAMI, A. G.; HERMANSKY, H. Segmentation of speech for speaker and language recognition. In: **Proc. 8th European Conference on Speech Communication Proceedings of the 8th European Conference on Speech Communication and Technology (Eurospeech 2003)**. [S.l.]: ISCA, 2003. p. 841—844. Citation on page 52.

AGGARWAL, C. **Data Streams: Models and Algorithms**. [S.l.]: Springer, 2007. (Advances in Database Systems). ISBN 9780387287591. Citation on page 115.

AGGARWAL, C. C.; WANG, H. **Managing and Mining Graph Data**. 1st. ed. [S.l.]: Springer Publishing Company, Incorporated, 2010. ISBN 1441960449, 9781441960443. Citation on page 93.

AGGARWAL, C. C.; ZHAO, Y.; YU, P. S. Outlier detection in graph streams. In: **Proceedings of the 2011 IEEE 27th International Conference on Data Engineering**. Washington, DC, USA: IEEE Computer Society, 2011. p. 399–409. ISBN 978-1-4244-8959-6. Citation on page 56.

AGHABOZORGI, S.; WAH, T. Y.; HERAWAN, T.; JALAB, H. A.; SHAYGAN, M. A.; JALALI, A. A hybrid algorithm for clustering of time series data based on affinity search technique. **The Scientific World Journal**, v. 2014, p. 12, 2014. Citation on page 44.

AGRAWAL, R.; GEHRKE, J.; GUNOPULOS, D.; RAGHAVAN, P. Automatic subspace clustering of high dimensional data for data mining applications. In: **Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data**. New York, NY, USA: ACM, 1998. (SIGMOD '98), p. 94–105. ISBN 0-89791-995-5. Citation on page 41.

AHA, D. W. (Ed.). **Lazy Learning**. Norwell, MA, USA: Kluwer Academic Publishers, 1997. ISBN 0-7923-4584-3. Citation on page 48.

AHMED, N. K.; ATIYA, A. F.; GAYAR, N. E.; EL-SHISHINY, H. An empirical comparison of machine learning models for time series forecasting. **Econometric Reviews**, v. 29, n. 5-6, p. 594–621, 2010. Citation on page 60.

AIELLO, W.; CHUNG, F.; LU, L. A random graph model for massive graphs. In: **Proceedings of the Thirty-second Annual ACM Symposium on Theory of Computing**. New York, NY, USA: ACM, 2000. (STOC '00), p. 171–180. ISBN 1-58113-184-4. Citation on page 28.

_____. Random evolution in massive graphs. In: ABELLO, J.; PARDALOS, P.; RESENDE, M. (Ed.). **Handbook of Massive Data Sets**. [S.l.]: Springer US, 2002, (Massive Computing, v. 4). p. 97–122. ISBN 978-1-4613-4882-5. Citation on page 28.

ALDECOA, R.; MARIN, I. Exploring the limits of community detection strategies in complex networks. **Sci. Rep.**, Macmillan Publishers Limited. All rights reserved, v. 3, 07 2013. Citation on page 71.

ALONSO, A.; BERRENDERO, J.; HERNÁNDEZ, A.; JUSTEL, A. Time series clustering based on forecast densities. **Computational Statistics and Data Analysis**, v. 51, n. 2, p. 762 – 776, 2006. ISSN 0167-9473. Citation on page 45.

ALONSO, C.; PRIETO, Ó.; RODRÍGUEZ, J.; BREGÓN, A. Multivariate time series classification via stacking of univariate classifiers. In: OKUN, O.; VALENTINI, G. (Ed.). **Supervised and Unsupervised Ensemble Methods and their Applications**. [S.l.]: Springer Berlin Heidelberg, 2008, (Studies in Computational Intelligence, v. 126). p. 135–151. ISBN 978-3-540-78980-2. Citation on page 50.

ALTMAN, N. S. An introduction to kernel and nearest-neighbor nonparametric regression. **The American Statistician**, 1992. Citations on pages 27 and 59.

AMARAL, L. A. N.; SCALA, A.; BARTHÉLÉMY, M.; STANLEY, H. E. Classes of small-world networks. **Proceedings of the National Academy of Sciences**, v. 97, n. 21, p. 11149–11152, 2000. Citation on page 28.

ANGIULLI, F.; FASSETTI, F. Detecting distance-based outliers in streams of data. In: **Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management**. New York, NY, USA: ACM, 2007. (CIKM '07), p. 811–820. ISBN 978-1-59593-803-9. Citation on page 56.

ANKERST, M.; BREUNIG, M. M.; KRIEGEL, H.-P.; SANDER, J. Optics: Ordering points to identify the clustering structure. In: **Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data**. New York, NY, USA: ACM, 1999. (SIGMOD '99), p. 49–60. ISBN 1-58113-084-8. Citation on page 41.

ARENAS, A.; DÍAZ-GUILERA, A.; PÉREZ-VICENTE, C. J. Synchronization reveals topological scales in complex networks. **Phys. Rev. Lett.**, American Physical Society, v. 96, p. 114102, Mar 2006. Citation on page 71.

ARIAS, M.; ARRATIA, A.; XURIGUERA, R. Forecasting with twitter data. **ACM Trans. Intell. Syst. Technol.**, ACM, New York, NY, USA, v. 5, n. 1, p. 8:1–8:24, Jan. 2014. ISSN 2157-6904. Citation on page 61.

ASSFALG, J.; KRIEGEL, H.-P.; KRÖGER PEERGER, P.; KUNATH, P.; PRYAKHIN, A.; RENZ, M. Similarity search on time series based on threshold queries. In: **Proceedings of the 10th International Conference on Advances in Database Technology**. Berlin, Heidelberg: Springer-Verlag, 2006. (EDBT06), p. 276–294. ISBN 3-540-32960-9, 978-3-540-32960-2. Citation on page 38.

BAKSHI, B.; STEPHANOPOULOS, G. Representation of process trends-iv. induction of real-time patterns from operating data for diagnosis and supervisory control. **Computers and Chemical Engineering**, v. 18, n. 4, p. 303 – 332, 1994. ISSN 0098-1354. An International Journal of Computer Applications in Chemical Engineering. Citation on page 28.

BARABÁSI, A.; JEONG, H.; NÉDA, Z.; RAVASZ, E.; SCHUBERT, A.; VICSEK, T. Evolution of the social network of scientific collaborations. **Physica A: Statistical Mechanics and its Applications**, v. 311, n. 3–4, p. 590 – 614, 2002. ISSN 0378-4371. Citation on page 28.

BARABASI, A. L.; ALBERT, R. Emergence of Scaling in Random Networks. **Science**, v. 286, n. 5439, p. 509–512, October 1999. Citations on pages 29 and 67.

BARRETO, G. A. Time series prediction with the self-organizing map: A review. In: HAMMER, B.; HITZLER, P. (Ed.). **Perspectives of Neural-Symbolic Integration**. [S.l.]: Springer Berlin Heidelberg, 2007, (Studies in Computational Intelligence, v. 77). p. 135–158. ISBN 978-3-540-73953-1. Citation on page 58.

- BASU, S.; MECKESHEIMER, M. Automatic outlier detection for time series: An application to sensor data. **Knowl. Inf. Syst.**, Springer-Verlag New York, Inc., New York, NY, USA, v. 11, n. 2, p. 137–154, Feb. 2007. ISSN 0219-1377. Citation on page 55.
- BATISTA, G. E. A. P. A.; KEOGH, E. J.; TATAW, O. M.; SOUZA, V. M. A. de. Cid: an efficient complexity-invariant distance for time series. **Data Mining and Knowledge Discovery**, Springer US, v. 28, n. 3, p. 634–669, 2014. ISSN 1384-5810. Citation on page 80.
- BERNDT, D. J.; CLIFFORD, J. Using dynamic time warping to find patterns in time series. In: FAYYAD, U. M.; UTHURUSAMY, R. (Ed.). **KDD Workshop**. [S.l.]: AAAI Press, 1994. p. 359–370. ISBN 0-929280-73-3. Citations on pages 38 and 80.
- BEZDEK, J. C. **Pattern Recognition with Fuzzy Objective Function Algorithms**. Norwell, MA, USA: Kluwer Academic Publishers, 1981. ISBN 0306406713. Citation on page 41.
- BISHOP, C. M. **Neural Networks for Pattern Recognition**. [S.l.]: Oxford University Press, Inc., 1995. Citation on page 27.
- BLONDEL, V. D.; GUILLAUME, J.-L.; LAMBIOTTE, R.; LEFEBVRE, E. Fast unfolding of communities in large networks. **Journal of Statistical Mechanics: Theory and Experiment**, v. 2008, n. 10, p. P10008, 2008. Citation on page 81.
- BLOOMFIELD, P. **Fourier Analysis of Time Series: An Introduction**. [S.l.]: Wiley, 2004. (Fourier Analysis of Time Series: An Introduction). Citation on page 27.
- BLUM, A. Empirical support for winnow and weighted-majority algorithms: Results on a calendar scheduling domain. **Mach. Learn.**, Kluwer Academic Publishers, Hingham, MA, USA, v. 26, n. 1, p. 5–23, Jan. 1997. ISSN 0885-6125. Citation on page 47.
- BOBER, M. Mpeg-7 visual shape descriptors. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 11, n. 6, p. 716–719, Jun 2001. ISSN 1051-8215. Citation on page 101.
- BOCCALETTI, S.; IVANCHENKO, M.; LATORA, V.; PLUCHINO, A.; RAPISARDA, A. Detecting complex network modularity by dynamical clustering. **Phys. Rev. E**, American Physical Society, v. 75, p. 045102, Apr 2007. Citation on page 71.
- BOCCALETTI, S.; LATORA, V.; MORENO, Y.; CHAVEZ, M.; HWANG, D.-U. Complex networks: Structure and dynamics. **Physics Reports**, v. 424, n. 4–5, p. 175 – 308, 2006. ISSN 0370-1573. Citations on pages 28, 31, and 72.
- BOCK, H.-H. Clustering Methods: A History of k-Means Algorithms. In: BRITO, P.; CUCUMEL, G.; BERTRAND, P.; CARVALHO, F. (Ed.). **Selected Contributions in Data Analysis and Classification**. [S.l.]: Springer Berlin Heidelberg, 2007, (Studies in Classification, Data Analysis, and Knowledge Organization). chap. 15, p. 161–172. Citation on page 41.
- BONANNO, G.; VANDEWALLE, N.; MANTEGNA, R. N. Taxonomy of stock market indices. **Phys. Rev. E**, American Physical Society, v. 62, p. R7615–R7618, Dec 2000. Citation on page 30.
- BONTEMPI, G.; TAIEB, S. B.; BORGNE, Y.-A. Machine learning strategies for time series forecasting. In: AUFAURE, M.-A.; ZIMÁNYI, E. (Ed.). **Business Intelligence**. [S.l.]: Springer Berlin Heidelberg, 2013, (Lecture Notes in Business Information Processing, v. 138). p. 62–77. ISBN 978-3-642-36317-7. Citation on page 59.

- BORGWARDT, K. M. **Graph kernels**. Phd Thesis (PhD Thesis) — lmu, 2007. Citation on page 96.
- BOX, G. E. P.; REINSEL, G. C.; JENKINS, G. M. **Time series analysis : forecasting and control**. [S.l.]: Prentice-Hall, 1994. Citations on pages 27 and 60.
- BRANDMAIER, A. M. **Permutation distribution clustering and structural equation model trees**. Phd Thesis (PhD Thesis) — Universität des Saarlandes, 2011. Citations on pages 85 and 87.
- BREIMAN, L.; FRIEDMAN, J. H.; OLSHEN, R. A.; STONE, C. J. **Classification and Regression Trees**. [S.l.]: Wadsworth International Group, 1984. Citations on pages 27 and 59.
- BROWN, A. E. X.; YEMINI, E. I.; GRUNDY, L. J.; JUCIKAS, T.; SCHAFER, W. R. A dictionary of behavioral motifs reveals clusters of genes affecting *caenorhabditis elegans* locomotion. **Proceedings of the National Academy of Sciences**, v. 110, n. 2, p. 791–796, 2013. Citation on page 58.
- BU, Y.; LEUNG, O. T.-W.; FU, A. W.-C.; KEOGH, E. J.; PEI, J.; MESHKIN, S. Wat: Finding top-k discords in time series database. In: **SDM**. [S.l.]: SIAM, 2007. p. 449–454. Citation on page 55.
- BUDALAKOTI, S.; BUDALAKOTI, S.; SRIVASTAVA, A.; OTEY, M.; OTEY, M. Anomaly detection and diagnosis algorithms for discrete symbol sequences with applications to airline safety. **Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on**, v. 39, n. 1, p. 101–113, Jan 2009. ISSN 1094-6977. Citation on page 54.
- BURDICK, D.; CALIMLIM, M.; GEHRKE, J. Mafia: a maximal frequent itemset algorithm for transactional databases. In: **Data Engineering, 2001. Proceedings. 17th International Conference on**. [S.l.: s.n.], 2001. p. 443–452. ISSN 1063-6382. Citation on page 41.
- BURGES, C. J. C. A tutorial on support vector machines for pattern recognition. **Data Min. Knowl. Discov.**, Kluwer Academic Publishers, Hingham, MA, USA, v. 2, n. 2, p. 121–167, Jun. 1998. ISSN 1384-5810. Citation on page 49.
- BURKOM, H. S.; MURPHY, S. P.; SHMUELI, G. Automated time series forecasting for biosurveillance. **Statistics in Medicine**, John Wiley & Sons, Ltd., v. 26, n. 22, p. 4202–4218, 2007. ISSN 1097-0258. Citation on page 27.
- CABRERA, J. a. B. D.; LEWIS, L.; MEHRA, R. K. Detection and classification of intrusions and faults using sequences of system calls. **SIGMOD Rec.**, ACM, New York, NY, USA, v. 30, n. 4, p. 25–34, Dec. 2001. ISSN 0163-5808. Citation on page 54.
- CAMPANA, B. J. L.; KEOGH, E. J. A compression based distance measure for texture. **Stat. Anal. Data Min.**, John Wiley & Sons, Inc., New York, NY, USA, v. 3, n. 6, p. 381–398, Dec. 2010. ISSN 1932-1864. Citation on page 38.
- CAMPANHARO, A. S. L. O.; SIRER, M. I.; MALMGREN, R. D.; RAMOS, F. M.; AMARAL, L. a. N. Duality between time series and networks. **PloS one**, v. 6, n. 8, p. e23378, Jan. 2011. ISSN 1932-6203. Citations on pages 31 and 94.
- CAPOCCI, A.; SERVEDIO, V.; CALDARELLI, G.; COLAIORI, F. Detecting communities in large networks. **Physica A: Statistical Mechanics and its Applications**, v. 352, n. 2–4, p. 669 – 676, 2005. ISSN 0378-4371. Citation on page 70.

CAPPÉ, O.; MOULINES, E.; RYDEN, T. **Inference in Hidden Markov Models**. [S.l.]: Springer-Verlag New York, Inc., 2005. (Springer Series in Statistics). Citation on page 27.

CARPENTER, G. A.; GROSSBERG, S. A massively parallel architecture for a self-organizing neural pattern recognition machine. **Comput. Vision Graph. Image Process.**, Academic Press Professional, Inc., San Diego, CA, USA, v. 37, n. 1, p. 54–115, Jan. 1987. ISSN 0734-189X. Citation on page 42.

CERRA, D.; BIENIARZ, J.; AVBELJ, J.; REINARTZ, P.; MUELLER, R. Compression-based unsupervised clustering of spectral signatures. In: **Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS), 2011 3rd Workshop on**. [S.l.: s.n.], 2011. p. 1–4. ISSN 2158-6268. Citation on page 45.

CHAKRABARTI, K.; KEOGH, E.; MEHROTRA, S.; PAZZANI, M. Locally adaptive dimensionality reduction for indexing large time series databases. **ACM Trans. Database Syst.**, ACM, New York, NY, USA, v. 27, n. 2, p. 188–228, Jun. 2002. ISSN 0362-5915. Citation on page 37.

CHANDOLA, V.; BANERJEE, A.; KUMAR, V. Anomaly detection. **ACM Computing Surveys**, v. 41, n. 3, p. 1–58, Jul. 2009. ISSN 03600300. Citation on page 52.

CHANDOLA, V.; MITHAL, V.; KUMAR, V. Comparative evaluation of anomaly detection techniques for sequence data. In: **Data Mining, 2008. ICDM '08. Eighth IEEE International Conference on**. [S.l.: s.n.], 2008. p. 743–748. ISSN 1550-4786. Citation on page 54.

CHANDRAKALA, S.; SEKHAR, C. A density based method for multivariate time series clustering in kernel feature space. In: **Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on**. [S.l.: s.n.], 2008. p. 1885–1890. ISSN 1098-7576. Citation on page 43.

CHEESEMAN, P.; STUTZ, J. Advances in knowledge discovery and data mining. In: FAYYAD, U. M.; PIATETSKY-SHAPIO, G.; SMYTH, P.; UTHURUSAMY, R. (Ed.). Menlo Park, CA, USA: American Association for Artificial Intelligence, 1996. chap. Bayesian Classification (AutoClass): Theory and Results, p. 153–180. ISBN 0-262-56097-6. Citation on page 42.

CHEN, G. H.; NIKOLOV, S.; SHAH, D. A latent source model for nonparametric time series classification. In: BURGESS, C. J. C.; BOTTOU, L.; GHAHRAMANI, Z.; WEINBERGER, K. Q. (Ed.). **NIPS**. [S.l.: s.n.], 2013. p. 1088–1096. Citation on page 50.

CHEN, J.; FANG, H.-r.; SAAD, Y. Fast approximate knn graph construction for high dimensional data via recursive lanczos bisection. **J. Mach. Learn. Res.**, JMLR.org, v. 10, p. 1989–2012, Dec. 2009. ISSN 1532-4435. Citation on page 78.

CHEN, J. R. Useful clustering outcomes from meaningful time series clustering. In: **Proceedings of the Sixth Australasian Conference on Data Mining and Analytics - Volume 70**. Darlinghurst, Australia, Australia: Australian Computer Society, Inc., 2007. (AusDM '07), p. 101–109. ISBN 978-1-920682-51-4. Citation on page 45.

CHEN, L.; NG, R. On the marriage of lp-norms and edit distance. In: **Proceedings of the Thirtieth International Conference on Very Large Data Bases - Volume 30**. [S.l.]: VLDB Endowment, 2004. (VLDB '04), p. 792–803. ISBN 0-12-088469-0. Citation on page 38.

CHEN, L.; ÖZSU, M. T.; ORIA, V. Robust and fast similarity search for moving object trajectories. In: **Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data**. New York, NY, USA: ACM, 2005. (SIGMOD '05), p. 491–502. ISBN 1-59593-060-4. Citation on page 38.

CHEN, X.; PETROUNIAS, I. Discovering temporal association rules: Algorithms, language and system. In: **Data Engineering, 2000. Proceedings. 16th International Conference on**. [S.l.: s.n.], 2000. p. 306–306. ISSN 1063-6382. Citation on page 115.

CHEN, Y.; KEOGH, E.; HU, B.; BEGUM, N.; BAGNALL, A.; MUEEN, A.; BATISTA, G. **The UCR Time Series Classification Archive**. 2015. <www.cs.ucr.edu/~eamonn/time_series_data/>. Citations on pages 78, 79, and 101.

CHEN, Y.; NASCIMENTO, M. A.; OOI, B. C.; TUNG, A. K. H. Spade: On shape-based pattern detection in streaming time series. In: CHIRKOVA, R.; DOGAC, A.; ÖZSU, M. T.; SELLIS, T. K. (Ed.). **ICDE**. [S.l.]: IEEE, 2007. p. 786–795. Citation on page 38.

CHENG, H.; TAN, P.-N.; GAO, J.; SCRIPPS, J. Multistep-ahead time series prediction. In: NG, W.-K.; KITSUREGAWA, M.; LI, J.; CHANG, K. (Ed.). **Advances in Knowledge Discovery and Data Mining**. [S.l.]: Springer Berlin Heidelberg, 2006, (Lecture Notes in Computer Science, v. 3918). p. 765–774. ISBN 978-3-540-33206-0. Citation on page 60.

CHESWICK, B.; BURCH, H.; BRANIGAN, S. Mapping and visualizing the internet. In: **In Proceedings of the 2000 USENIX Annual Technical Conference**. [S.l.: s.n.], 2000. p. 1–12. Citation on page 29.

CHIU, B.; KEOGH, E.; LONARDI, S. Probabilistic discovery of time series motifs. **Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '03**, ACM Press, New York, New York, USA, p. 493, 2003. Citations on pages 57 and 58.

CHOUAKRIA, A. D.; NAGABHUSHAN, P. N. Adaptive dissimilarity index for measuring time series proximity. **Adv. Data Analysis and Classification**, v. 1, n. 1, p. 5–21, 2007. Citation on page 45.

CLAUSET, A.; NEWMAN, M. E. J.; MOORE, C. Finding community structure in very large networks. **Phys. Rev. E**, American Physical Society, v. 70, p. 066111, Dec 2004. Citations on pages 18, 30, 68, 71, 81, 109, and 110.

COHEN, W. W. Fast Effective Rule Induction. In: **In Proceedings of the Twelfth International Conference on Machine Learning**. [S.l.: s.n.], 1995. p. 115–123. Citation on page 46.

CONGDON, P. **Bayesian Statistical Modelling**. [S.l.]: Wiley, 2007. (Wiley Series in Probability and Statistics). ISBN 9780470035931. Citation on page 42.

CONTE, D.; FOGGIA, P.; SANSONE, C.; VENTO, M. Thirty years of graph matching in pattern recognition. **International Journal of Pattern Recognition and Artificial Intelligence**, v. 18, n. 03, p. 265–298, 2004. Citation on page 96.

COOK, D. J.; HOLDER, L. B. **Mining Graph Data**. [S.l.]: John Wiley & Sons, 2006. ISBN 0471731900. Citation on page 32.

CORTES, C.; VAPNIK, V. Support-vector networks. **Mach. Learn.**, Kluwer Academic Publishers, Hingham, MA, USA, v. 20, n. 3, p. 273–297, Sep. 1995. ISSN 0885-6125. Citation on page 48.

COSTA, L.; RODRIGUES, F. Characterization of complex networks: A survey of measurements. **Advances in Physics**, v. 56, n. 1, p. 167–242, January 2007. Citation on page 97.

COSTA, L. d. F.; RODRIGUES, F. A.; TRAVIESO, G.; BOAS, P. R. V. Characterization of complex networks: A survey of measurements. **Advances in Physics**, v. 56, n. 1, p. 167–242, 2007. Citations on pages 31 and 67.

COTOFREI, P.; STOFFEL, K. Classification rules + time = temporal rules. In: SLOOT, P.; HOEKSTRA, A.; TAN, C.; DONGARRA, J. (Ed.). **Computational Science — ICCS 2002**. [S.l.]: Springer Berlin Heidelberg, 2002, (Lecture Notes in Computer Science, v. 2329). p. 572–581. ISBN 978-3-540-43591-4. Citation on page 44.

CUARESMA, J. C.; HLOUSKOVA, J.; KOSSMEIER, S.; OBERSTEINER, M. Forecasting electricity spot-prices using linear univariate time-series models. **Applied Energy**, v. 77, n. 1, p. 87 – 106, 2004. ISSN 0306-2619. Citation on page 27.

CUPERTINO, T. H.; HUERTAS, J.; ZHAO, L. Data clustering using controlled consensus in complex networks. **Neurocomput.**, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 118, p. 132–140, Oct. 2013. ISSN 0925-2312. Citations on pages 30 and 71.

DASGUPTA, D.; NINO, F. A comparison of negative and positive selection algorithms in novel pattern detection. In: **Systems, Man, and Cybernetics, 2000 IEEE International Conference on**. [S.l.: s.n.], 2000. v. 1, p. 125–130 vol.1. ISSN 1062-922X. Citation on page 54.

DELLNITZ, M.; MOLO, M. H. von; METZNER, P.; PREIS, R.; SCHÜTTE, C. Graph Algorithms for Dynamical Systems. In: MIELKE, A. (Ed.). **Analysis, Modeling and Simulation of Multiscale Problems**. [S.l.]: Springer Berlin Heidelberg, 2006. p. 619–645. ISBN 9783540356578. Citation on page 74.

DEMPSTER, A. P.; LAIRD, N. M.; RUBIN, D. B. Maximum Likelihood from Incomplete Data via the EM Algorithm. **Journal of the Royal Statistical Society. Series B (Methodological)**, Blackwell Publishing for the Royal Statistical Society, v. 39, n. 1, p. 1–38, 1977. Citation on page 42.

DEMŠAR, J. Statistical comparisons of classifiers over multiple data sets. **J. Mach. Learn. Res.**, JMLR.org, v. 7, p. 1–30, Dec. 2006. ISSN 1532-4435. Citations on pages 80, 82, 84, and 86.

DENG, H.; RUNGER, G.; TUV, E.; VLADIMIR, M. A time series forest for classification and feature extraction. **Information Sciences**, v. 239, n. 0, p. 142 – 153, 2013. ISSN 0020-0255. Citation on page 50.

DENTON, A. Density-based clustering of time series subsequences. In: **Proceedings of The Third Workshop on Mining Temporal and Sequential Data (TDM 04) in conjunction with The Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. [S.l.: s.n.], 2004. Citation on page 45.

DIESTEL, R. **Graph Theory**. Third. [S.l.]: Springer-Verlag, Heidelberg, 2005. (Graduate Texts in Mathematics, v. 173). Citation on page 63.

DIMITRIADOU, E.; BARTH, M.; WINDISCHBERGER, C.; HORNIK, K.; MOSER, E. A quantitative comparison of functional {MRI} cluster analysis. **Artificial Intelligence in Medicine**, v. 31, n. 1, p. 57 – 71, 2004. ISSN 0933-3657. Citation on page 43.

DING, H.; TRAJCEVSKI, G.; SCHEUERMANN, P.; WANG, X.; KEOGH, E. Querying and mining of time series data: Experimental comparison of representations and distance measures. **Proc. VLDB Endow.**, VLDB Endowment, v. 1, n. 2, p. 1542–1552, Aug. 2008. ISSN 2150-8097. Citations on pages 38 and 114.

DONETTI, L.; MUÑOZ, M. A. Detecting network communities: a new systematic and efficient algorithm. **Journal of Statistical Mechanics: Theory and Experiment**, v. 2004, n. 10, p. P10012, 2004. Citations on pages 30, 69, and 70.

DONNER, R. V.; SMALL, M.; DONGES, J. F.; MARWAN, N.; ZOU, Y.; XIANG, R.; KURTHS, J. Recurrence-based time series analysis by means of complex network methods. **International Journal of Bifurcation and Chaos**, v. 21, n. 04, p. 1019–1046, 2011. Citations on pages 28, 72, 94, and 105.

Donner, R. V.; Zou, Y.; Donges, J. F.; Marwan, N.; Kurths, J. Complex networks approach to geophysical time series analysis: Detecting paleoclimate transitions via recurrence networks. **AGU Fall Meeting Abstracts**, p. A1183, Dec. 2009. Citation on page 31.

DONNER, R. V.; ZOU, Y.; DONGES, J. F.; MARWAN, N.; KURTHS, J. Recurrence networks—a novel paradigm for nonlinear time series analysis. **New Journal of Physics**, v. 12, n. 3, p. 033025, 2010. Citation on page 31.

DRUCKER, H.; BURGESS, C. J.; KAUFMAN, L.; C, C. J.; KAUFMAN, B. L.; SMOLA, A.; VAPNIK, V. Support vector regression machines. 1996. Citation on page 27.

DUNCAN, S.; BRYANT, G. A new algorithm for segmenting data from time series. In: **Decision and Control, 1996., Proceedings of the 35th IEEE Conference on**. [S.l.: s.n.], 1996. v. 3, p. 3123–3128 vol.3. ISSN 0191-2216. Citation on page 52.

DŽEROSKI, S. Multi-relational data mining: An introduction. **SIGKDD Explor. Newsl.**, ACM, New York, NY, USA, v. 5, n. 1, p. 1–16, Jul. 2003. ISSN 1931-0145. Citations on pages 32 and 35.

EADS, D. R.; HILL, D.; DAVIS, S.; PERKINS, S. J.; MA, J.; PORTER, R. B.; THEILER, J. P. Genetic algorithms and support vector machines for time series classification. **Proc. SPIE**, v. 4787, p. 74–85, 2002. Citation on page 49.

EBEL, H.; MIELSCH, L.-I.; BORNHOLDT, S. Scale-free topology of e-mail networks. **Phys. Rev. E**, American Physical Society, v. 66, p. 035103, Sep 2002. Citation on page 28.

EDWARDS, R.; MAGEE, J.; BASSETTI, W. **Technical Analysis of Stock Trends**. [S.l.]: Taylor & Francis, 2007. ISBN 9781420013146. Citation on page 115.

ELFEKY, M.; AREF, W.; ELMAGARMID, A. Periodicity detection in time series databases. **Knowledge and Data Engineering, IEEE Transactions on**, v. 17, n. 7, p. 875–887, July 2005. ISSN 1041-4347. Citation on page 106.

_____. Warp: time warping for periodicity detection. In: **Data Mining, Fifth IEEE International Conference on**. [S.l.: s.n.], 2005. p. 8 pp.–. ISSN 1550-4786. Citation on page 106.

ELSNER, J. B.; JAGGER, T. H.; FOGARTY, E. A. Visibility network of united states hurricanes. **Geophysical Research Letters**, v. 36, n. 16, 2009. ISSN 1944-8007. Citation on page 74.

ERDÖS, P.; RÉNYI, A. On random graphs, I. **Publicationes Mathematicae (Debrecen)**, 1959. Citations on pages 29 and 65.

ESLING, P.; AGON, C. Time-series data mining. **ACM Comput. Surv.**, ACM, New York, NY, USA, v. 45, n. 1, p. 12:1–12:34, Dec. 2012. ISSN 0360-0300. Citations on pages 28, 38, 39, 42, 50, 51, 53, 57, 59, and 75.

ESTER, M.; KRIEGEL, H. P.; SANDER, J.; XU, X. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In: SIMOUDIS, E.; HAN, J.; FAYYAD, U. (Ed.). **Second International Conference on Knowledge Discovery and Data Mining**. Portland, Oregon: AAAI Press, 1996. p. 226–231. Citation on page 41.

FALOUTSOS, C.; RANGANATHAN, M.; MANOLOPOULOS, Y. Fast subsequence matching in time-series databases. In: **Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data**. New York, NY, USA: ACM, 1994. (SIGMOD '94), p. 419–429. ISBN 0-89791-639-5. Citations on pages 28 and 38.

FALOUTSOS, M.; FALOUTSOS, P.; FALOUTSOS, C. On power-law relationships of the internet topology. **SIGCOMM Comput. Commun. Rev.**, ACM, New York, NY, USA, v. 29, n. 4, p. 251–262, Aug. 1999. ISSN 0146-4833. Citation on page 28.

FERREIRA, P.; AZEVEDO, P.; SILVA, C.; BRITO, R. Mining approximate motifs in time series. In: TODOROVSKI, L.; LAVRAC, N.; JANTKE, K. (Ed.). **Discovery Science**. [S.l.]: Springer Berlin Heidelberg, 2006, (Lecture Notes in Computer Science, v. 4265). p. 89–101. ISBN 978-3-540-46491-4. Citations on pages 57 and 58.

FITZGIBBON, L.; DOWE, D.; ALLISON, L. Change-point estimation using new minimum message length approximations. In: ISHIZUKA, M.; SATTAR, A. (Ed.). **PRICAI 2002: Trends in Artificial Intelligence**. [S.l.]: Springer Berlin Heidelberg, 2002, (Lecture Notes in Computer Science, v. 2417). p. 244–254. ISBN 978-3-540-44038-3. Citation on page 52.

FLAKE, G.; LAWRENCE, S.; GILES, C.; COETZEE, F. Self-organization and identification of web communities. **Computer**, v. 35, n. 3, p. 66–70, Mar 2002. ISSN 0018-9162. Citation on page 69.

FORGY, E. W. Cluster analysis of multivariate data: efficiency vs interpretability of classifications. **Biometrics**, v. 21, p. 768–769, 1965. Citation on page 41.

FORTUNATO, S. Community detection in graphs. **Physics Reports**, v. 486, n. 3–5, p. 75 – 174, 2010. ISSN 0370-1573. Citations on pages 30, 32, 69, and 72.

FRANK, E.; WITTEN, I. H. Generating accurate rule sets without global optimization. In: **Proceedings of the Fifteenth International Conference on Machine Learning**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998. (ICML '98), p. 144–151. ISBN 1-55860-556-8. Citation on page 46.

FRENTZOS, E.; GRATSIAS, K.; THEODORIDIS, Y. Index-based most similar trajectory search. In: **Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on**. [S.l.: s.n.], 2007. p. 816–825. Citations on pages 38 and 80.

FRIEDMAN, N.; GEIGER, D.; GOLDSZMIDT, M. Bayesian network classifiers. **Mach. Learn.**, Kluwer Academic Publishers, Hingham, MA, USA, v. 29, n. 2-3, p. 131–163, Nov. 1997. ISSN 0885-6125. Citation on page 47.

FU, T.-c. A review on time series data mining. **Engineering Applications of Artificial Intelligence**, Elsevier, v. 24, n. 1, p. 164–181, Feb. 2011. ISSN 09521976. Citations on pages 28, 50, and 52.

FU, T.-C.; CHUNG, F.-L.; NG, C. man. Financial time series segmentation based on specialized binary tree representation. In: CRONE, S. F.; LESSMANN, S.; STAHLBOCK, R. (Ed.). **DMIN**. [S.l.]: CSREA Press, 2006. p. 3–9. ISBN 1-60132-004-3. Citation on page 52.

GABER, M. M.; ZASLAVSKY, A.; KRISHNASWAMY, S. Mining data streams: A review. **SIGMOD Rec.**, ACM, New York, NY, USA, v. 34, n. 2, p. 18–26, Jun. 2005. ISSN 0163-5808. Citation on page 115.

GAMA, J. **Knowledge Discovery from Data Streams**. 1st. ed. [S.l.]: Chapman & Hall/CRC, 2010. ISBN 1439826110, 9781439826119. Citation on page 115.

GAN, G.; MA, C.; WU, J. **Data Clustering: Theory, Algorithms, and Applications**. [S.l.]: Society for Industrial and Applied Mathematics, 2007. Citation on page 85.

GAO, B.; MA, H.-Y.; YANG, Y.-H. Hmms (hidden markov models) based on anomaly intrusion detection method. In: **Machine Learning and Cybernetics, 2002. Proceedings. 2002 International Conference on**. [S.l.: s.n.], 2002. v. 1, p. 381–385 vol.1. Citation on page 54.

GAO, J.; SULTAN, H.; HU, J.; TUNG, W. wen. Denoising nonlinear time series by adaptive filtering and wavelet shrinkage: A comparison. **Signal Processing Letters, IEEE**, v. 17, n. 3, p. 237–240, March 2010. ISSN 1070-9908. Citation on page 37.

GAO, X.; XIAO, B.; TAO, D.; LI, X. A survey of graph edit distance. **Pattern Analysis and Applications**, Springer-Verlag, v. 13, n. 1, p. 113–129, 2010. ISSN 1433-7541. Citation on page 96.

GARCIA, S.; LUENGO, J.; SÁEZ, J.; LÓPEZ, V.; HERRERA, F. A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning. **Knowledge and Data Engineering, IEEE Transactions on**, v. 25, n. 4, p. 734–750, April 2013. ISSN 1041-4347. Citation on page 107.

GARGI, U.; LU, W.; MIRROKNI, V.; YOON, S. Large-scale community detection on youtube for topic discovery and exploration. In: **in Proc. of the Fifth international AAAI Conference on Weblogs and Social Media**. [S.l.: s.n.], 2011. p. 486–489. Citation on page 72.

GE, Y.; XIONG, H.; ZHOU, Z.-h.; OZDEMIR, H.; YU, J.; LEE, K. C. Top-eye: Top-k evolving trajectory outlier detection. In: **Proceedings of the 19th ACM International Conference on Information and Knowledge Management**. New York, NY, USA: ACM, 2010. (CIKM '10), p. 1733–1736. ISBN 978-1-4503-0099-5. Citation on page 56.

GETOOR, L. Link mining: A new data mining challenge. **SIGKDD Explor. Newsl.**, ACM, New York, NY, USA, v. 5, n. 1, p. 84–89, Jul. 2003. ISSN 1931-0145. Citations on pages 31 and 32.

GEURTS, P. **Contributions to Decision Tree Induction: Bias/Variance Tradeoff and Time Series Classification**. Phd Thesis (PhD Thesis) — Department of Electrical Engineering, University of Liege, Belgium, 2002. Citations on pages 88 and 90.

GHOSH, A. K.; SCHWARTZBARD, A.; SCHATZ, M. Learning program behavior profiles for intrusion detection. In: **Proceedings of the 1st Conference on Workshop on Intrusion Detection and Network Monitoring - Volume 1**. Berkeley, CA, USA: USENIX Association, 1999. (ID'99), p. 6–6. Citation on page 54.

GHOSH, S. **Visibility Algorithms in the Plane**. New York, NY, USA: Cambridge University Press, 2007. ISBN 0521875749. Citation on page 95.

GIRVAN, M.; NEWMAN, M. E. J. Community structure in social and biological networks. **Proceedings of the National Academy of Sciences**, v. 99, n. 12, p. 7821–7826, 2002. Citations on pages 30, 69, and 70.

GOLAY, X.; KOLLIAS, S.; STOLL, G.; MEIER, D.; VALAVANIS, a.; BOESIGER, P. A new correlation-based fuzzy logic clustering algorithm for fMRI. **Magnetic resonance in medicine : official journal of the Society of Magnetic Resonance in Medicine / Society of Magnetic Resonance in Medicine**, v. 40, n. 2, p. 249–60, Aug. 1998. ISSN 0740-3194. Citations on pages 43 and 80.

GONZÁLEZ, F. A.; DASGUPTA, D. Anomaly detection using real-valued negative selection. **Genetic Programming and Evolvable Machines**, Kluwer Academic Publishers, Hingham, MA, USA, v. 4, n. 4, p. 383–403, Dec. 2003. ISSN 1389-2576. Citation on page 54.

GORR, W.; OLLIGSCHLAEGER, A.; THOMPSON, Y. Short-term forecasting of crime. **International Journal of Forecasting**, v. 19, n. 4, p. 579 – 594, 2003. ISSN 0169-2070. Citation on page 27.

GUHA, S.; RASTOGI, R.; SHIM, K. Cure: An efficient clustering algorithm for large databases. In: **Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data**. New York, NY, USA: ACM, 1998. (SIGMOD '98), p. 73–84. ISBN 0-89791-995-5. Citation on page 40.

GUO, C.; JIA, H.; ZHANG, N. Time series clustering based on ica for stock data analysis. In: **Wireless Communications, Networking and Mobile Computing, 2008. WiCOM '08. 4th International Conference on**. [S.l.: s.n.], 2008. p. 1–4. Citation on page 44.

GUPTA, M.; AGGARWAL, C.; HAN, J.; SUN, Y. Evolutionary clustering and analysis of bibliographic networks. In: **Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on**. [S.l.: s.n.], 2011. p. 63–70. Citation on page 56.

GUPTA, M.; AGGARWAL, C. C.; HAN, J. Finding top-k shortest path distance changes in an evolutionary network. In: **Proceedings of the 12th International Conference on Advances in Spatial and Temporal Databases**. Berlin, Heidelberg: Springer-Verlag, 2011. (SSTD'11), p. 130–148. ISBN 978-3-642-22921-3. Citation on page 55.

GUPTA, M.; GAO, J.; AGGARWAL, C. C.; HAN, J. Outlier detection for temporal data: A survey. **IEEE Transactions on Knowledge and Data Engineering**, IEEE Computer Society, Los Alamitos, CA, USA, v. 99, n. PrePrints, p. 1, 2013. ISSN 1041-4347. Citations on pages 52 and 53.

GUPTA, M.; GAO, J.; SUN, Y.; HAN, J. Integrating community matching and outlier detection for mining evolutionary community outliers. In: **Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: ACM, 2012. (KDD '12), p. 859–867. ISBN 978-1-4503-1462-6. Citation on page 56.

GURALNIK, V.; SRIVASTAVA, J. Event detection from time series data. In: **Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: ACM, 1999. (KDD '99), p. 33–42. ISBN 1-58113-143-7. Citation on page 52.

GWADERA, R.; ATALLAH, M.; SZPANKOWSKI, W. Reliable detection of episodes in event sequences. In: **Data Mining, 2003. ICDM 2003. Third IEEE International Conference on**. [S.l.: s.n.], 2003. p. 67–74. Citation on page 54.

HALKIDI, M.; BATISTAKIS, Y.; VAZIRGIANNIS, M. On clustering validation techniques. **J. Intell. Inf. Syst.**, Kluwer Academic Publishers, Hingham, MA, USA, v. 17, n. 2-3, p. 107–145, Dec. 2001. ISSN 0925-9902. Citation on page 80.

HAMILTON, J. D. **Time Series Analysis**. [S.l.]: Princeton University Press, 1994. Citation on page 60.

HAMZAÇEBİ, C.; AKAY, D.; KUTAY, F. Comparison of direct and iterative artificial neural network forecast approaches in multi-periodic time series forecasting. **Expert Syst. Appl.**, Pergamon Press, Inc., Tarrytown, NY, USA, v. 36, n. 2, p. 3839–3844, Mar. 2009. ISSN 0957-4174. Citation on page 60.

HAN, J.; GONG, W.; YIN, Y. Mining segment-wise periodic patterns in time-related databases. In: AGRAWAL, R.; STOLORZ, P. E.; PIATETSKY-SHAPIO, G. (Ed.). **KDD**. [S.l.]: AAAI Press, 1998. p. 214–218. ISBN 1-57735-070-7. Citation on page 106.

HAN, J.; KAMBER, M.; PEI, J. **Data Mining: Concepts and Techniques**. 3rd. ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011. ISBN 0123814790, 9780123814791. Citation on page 39.

HARTLEY, H. O. Maximum Likelihood Estimation from Incomplete Data. **Biometrics**, International Biometric Society, v. 14, n. 2, p. 174–194, 1958. Citation on page 42.

HARVEY, A. C. **Forecasting, Structural Time Series Models and the Kalman Filter**. [S.l.]: Cambridge University Press, 1990. Citation on page 27.

HAWKINS, D. **Identification of outliers**. [S.l.]: Chapman and Hall, 1980. ISBN 9780412219009. Citation on page 52.

HAYKIN, S. **Neural Networks and Learning Machines**. [S.l.]: Prentice Hall, 2009. (Neural networks and learning machines, v. 10). ISBN 9780131471399. Citations on pages 42 and 47.

HEBRIL, G.; HUGUENEY, B. Symbolic representation of long time-series. In: **Symbolic Data Analysis at the 4th European Conference on Principles of Data Mining and Knowledge Discovery**. [S.l.: s.n.], 2000. p. 56–65. Citation on page 45.

HERRERA, L. J.; POMARES, H.; ROJAS, I.; GUILLÉN, A.; PRIETO, A.; VALENZUELA, O. Recursive prediction for long term time series forecasting using advanced models. **Neurocomput.**, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 70, n. 16-18, p. 2870–2880, Oct. 2007. ISSN 0925-2312. Citation on page 59.

HIDASI, B.; GÁSPÁR-PAPANEK, C. Shifttree: An interpretable model-based approach for time series classification. In: GUNOPULOS, D.; HOFMANN, T.; MALERBA, D.; VAZIRGIANNIS, M. (Ed.). **ECML/PKDD (2)**. [S.l.]: Springer, 2011. (Lecture Notes in Computer Science, v. 6912), p. 48–64. ISBN 978-3-642-23782-9. Citation on page 50.

HILL, D. J.; MINSKER, B. S. Anomaly detection in streaming environmental sensor data: A data-driven modeling approach. **Environ. Model. Softw.**, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 25, n. 9, p. 1014–1022, Sep. 2010. ISSN 1364-8152. Citations on pages 55 and 56.

HINNEBURG, A.; KEIM, D. A. An efficient approach to clustering in large multimedia databases with noise. In: AGRAWAL, R.; STOLORZ, P. E.; PIATETSKY-SHAPIO, G. (Ed.). **KDD**. [S.l.]: AAAI Press, 1998. p. 58–65. ISBN 1-57735-070-7. Citation on page 41.

HOFMAN, J. M.; WIGGINS, C. H. Bayesian approach to network modularity. **Phys. Rev. Lett.**, American Physical Society, v. 100, p. 258701, Jun 2008. Citation on page 71.

HOLME, P.; SARAMÄKI, J. Temporal networks. **Physics Reports**, v. 519, n. 3, p. 97–125, 2012. Citation on page 55.

HOPFIELD, J. J. Neurocomputing: Foundations of research. In: ANDERSON, J. A.; ROSENFELD, E. (Ed.). Cambridge, MA, USA: MIT Press, 1988. chap. Neural Networks and Physical Systems with Emergent Collective Computational Abilities, p. 457–464. ISBN 0-262-01097-6. Citation on page 47.

HSU, C.-W.; LIN, C.-J. A comparison of methods for multiclass support vector machines. **Neural Networks, IEEE Transactions on**, v. 13, n. 2, p. 415–425, Mar 2002. ISSN 1045-9227. Citation on page 49.

HUANG, Z. A Fast Clustering Algorithm to Cluster Very Large Categorical Data Sets in Data Mining. In: **SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery**. [S.l.: s.n.], 1997. p. 1–8. Citation on page 41.

HUBERMAN, B. **The Laws of the Web: Patterns in the Ecology of Information**. [S.l.]: MIT Press, 2003. ISBN 9780262250412. Citation on page 28.

HULTEN, G.; SPENCER, L.; DOMINGOS, P. Mining time-changing data streams. In: **Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: ACM, 2001. (KDD '01), p. 97–106. ISBN 1-58113-391-X. Citation on page 115.

HÜSKEN, M.; STAGGE, P. Recurrent neural networks for time series classification. **Neurocomputing**, v. 50, n. 0, p. 223 – 235, 2003. ISSN 0925-2312. Citation on page 50.

HYAFIL, L.; RIVEST, R. L. Constructing optimal binary decision trees is np-complete. **Information Processing Letters**, v. 5, n. 1, p. 15 – 17, 1976. ISSN 0020-0190. Citation on page 46.

HYVÄRINEN, A.; KARHUNEN, J.; OJA, E. **Independent Component Analysis**. [S.l.]: Wiley, 2004. (Adaptive and Learning Systems for Signal Processing, Communications and Control Series). ISBN 9780471464198. Citation on page 44.

IDÉ, T.; KASHIMA, H. Eigenspace-based anomaly detection in computer systems. In: **Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: ACM, 2004. (KDD '04), p. 440–449. ISBN 1-58113-888-1. Citation on page 56.

INMET - Instituto Nacional de Meteorologia. **Historical average monthly temperatures from the city of São Carlos - SP, Brazil**. 2013. [Http://www.inmet.gov.br/](http://www.inmet.gov.br/). Citation on page 110.

JAIN, A. K.; MURTY, M. N.; FLYNN, P. J. Data clustering: A review. **ACM Comput. Surv.**, ACM, New York, NY, USA, v. 31, n. 3, p. 264–323, Sep. 1999. ISSN 0360-0300. Citation on page 43.

JENSEN, F. V.; NIELSEN, T. D. **Bayesian Networks and Decision Graphs**. 2nd. ed. [S.l.]: Springer Publishing Company, Incorporated, 2007. ISBN 9780387682815. Citation on page 48.

JOVIĆ, A.; BRKIĆ, K.; BOGUNOVIĆ, N. Decision tree ensembles in biomedical time-series classification. In: PINZ, A.; POCK, T.; BISCHOF, H.; LEBERL, F. (Ed.). **Pattern Recognition**. [S.l.]: Springer Berlin Heidelberg, 2012, (Lecture Notes in Computer Science, v. 7476). p. 408–417. ISBN 978-3-642-32716-2. Citation on page 50.

KAMATH, C.; FAN, Y. J. Finding motifs in wind generation time series data. In: **Machine Learning and Applications (ICMLA), 2012 11th International Conference on**. [S.l.: s.n.], 2012. v. 2, p. 481–486. Citation on page 58.

KAMPOURAKI, A.; MANIS, G.; NIKOU, C. Heartbeat time series classification with support vector machines. **Information Technology in Biomedicine, IEEE Transactions on**, v. 13, n. 4, p. 512–518, July 2009. ISSN 1089-7771. Citation on page 50.

KARYPIS, G.; HAN, E.-H. S.; KUMAR, V. Chameleon: Hierarchical clustering using dynamic modeling. **Computer**, IEEE Computer Society Press, Los Alamitos, CA, USA, v. 32, n. 8, p. 68–75, Aug. 1999. ISSN 0018-9162. Citation on page 40.

KAUFMAN, L.; ROUSSEEUW, P. **Clustering by Means of Medoids**. [S.l.]: Faculty of Mathematics and Informatics, 1987. (Reports of the Faculty of Mathematics and Informatics). Citation on page 41.

_____. **Finding groups in data: an introduction to cluster analysis**. [S.l.]: Wiley, 2005. (Wiley series in probability and mathematical statistics. Applied probability and statistics). ISBN 9780471735786. Citation on page 41.

KEOGH, E.; CHU, S.; HART, D.; PAZZANI, M. Segmenting time series: A survey and novel approach. In: **In an Edited Volume, Data mining in Time Series Databases. Published by World Scientific**. [S.l.]: Publishing Company, 1993. p. 1–22. Citations on pages 28, 50, and 51.

KEOGH, E.; LIN, J.; FU, A. Hot sax: efficiently finding the most unusual time series subsequence. In: **Data Mining, Fifth IEEE International Conference on**. [S.l.: s.n.], 2005. p. 8. ISSN 1550-4786. Citation on page 56.

KEOGH, E.; LIN, J.; LEE, S.-H.; HERLE, H. V. Finding the most unusual time series subsequence: Algorithms and applications. **Knowl. Inf. Syst.**, Springer-Verlag New York, Inc., New York, NY, USA, v. 11, n. 1, p. 1–27, Dec. 2006. ISSN 0219-1377. Citation on page 55.

KEOGH, E.; LIN, J.; TRUPPEL, W. Clustering of time series subsequences is meaningless: implications for previous and future research. In: **Data Mining, 2003. ICDM 2003. Third IEEE International Conference on**. [S.l.: s.n.], 2003. p. 115–122. Citations on pages 33 and 45.

KEOGH, E.; LONARDI, S.; CHIU, B. Y.-c. Finding surprising patterns in a time series database in linear time and space. In: **Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: ACM, 2002. (KDD '02), p. 550–556. ISBN 1-58113-567-X. Citation on page 54.

KEOGH, E.; PAZZANI, M. An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. In: AGRAWAL, R.; STOLORZ, P.; PIATETSKY-SHAPIO, G. (Ed.). **Fourth International Conference on Knowledge Discovery and Data Mining (KDD'98)**. New York City, NY: ACM Press, 1998. p. 239–241. Citation on page 51.

KEOGH, E. J.; PAZZANI, M. J. Relevance feedback retrieval of time series data. In: **Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval**. New York, NY, USA: ACM, 1999. (SIGIR '99), p. 183–190. ISBN 1-58113-096-1. Citation on page 51.

KEOGH, E. J.; SMYTH, P. A probabilistic approach to fast pattern matching in time series databases. In: **KDD**. [S.l.: s.n.], 1997. p. 24–30. Citation on page 51.

KHAN, L. Data stream mining: Challenges and techniques. In: **Tools with Artificial Intelligence (ICTAI), 2010 22nd IEEE International Conference on**. [S.l.: s.n.], 2010. v. 2, p. 295–295. ISSN 1082-3409. Citation on page 115.

KOHONEN, T. The self-organizing map. **Proceedings of the IEEE**, v. 78, n. 9, p. 1464–1480, Sep 1990. ISSN 0018-9219. Citation on page 42.

_____. **Self-Organizing Maps**. [S.l.]: Springer Berlin Heidelberg, 2001. (Physics and astronomy online library). ISBN 9783540679219. Citation on page 42.

KOTSIANTIS, S. B. Supervised machine learning: A review of classification techniques. In: **Proceedings of the 2007 Conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies**. Amsterdam, The Netherlands, The Netherlands: IOS Press, 2007. p. 3–24. ISBN 978-1-58603-780-2. Citations on pages 46 and 49.

KRAUSE, A. E.; FRANK, K. A.; MASON, D. M.; ULANOWICZ, R. E.; TAYLOR, W. W. Compartments revealed in food-web structure. **Nature**, v. 426, n. 6964, p. 282–285, 11 2003. Citation on page 69.

KREMER, H.; GUNNEMANN, S.; SEIDL, T. Detecting climate change in multivariate time series data by novel clustering and cluster tracing techniques. In: **Data Mining Workshops (ICDMW), 2010 IEEE International Conference on**. [S.l.: s.n.], 2010. p. 96–97. Citation on page 45.

KRISHNAPURAM, R.; JOSHI, A.; YI, L. A fuzzy relative of the k-medoids algorithm with application to web document and snippet clustering. In: **Fuzzy Systems Conference Proceedings, 1999. FUZZ-IEEE '99. 1999 IEEE International**. [S.l.: s.n.], 1999. v. 3, p. 1281–1286 vol.3. ISSN 1098-7584. Citation on page 41.

LACASA, L.; LUQUE, B.; BALLESTEROS, F.; LUQUE, J.; NUÑO, J. C. From time series to complex networks: The visibility graph. **Proceedings of the National Academy of Sciences**, v. 105, n. 13, p. 4972–4975, 2008. Citations on pages 31, 73, and 94.

LANE, T.; BRODLEY, C. E. Sequence Matching and Learning in Anomaly Detection for Computer Security. In: FAWCETT; HAIMOWITZ; PROVOST; STOLFO (Ed.). **AI Approaches to Fraud Detection and Risk Management**. [S.l.]: AAAI Press, 1997. p. 43–49. Citation on page 54.

_____. Temporal sequence learning and data reduction for anomaly detection. **ACM Trans. Inf. Syst. Secur.**, ACM, New York, NY, USA, v. 2, n. 3, p. 295–331, Aug. 1999. ISSN 1094-9224. Citation on page 54.

LAVRENKO, V.; SCHMILL, M.; LAWRIE, D.; OGILVIE, P.; JENSEN, D.; ALLAN, J. Mining of Concurrent Text and Time Series. In: **KDD-2000 Workshop on Text Mining**. [S.l.: s.n.], 2000. p. 37–44. Citation on page 51.

LAXMAN, S.; SASTRY, P. A survey of temporal data mining. **Sadhana**, Springer India, v. 31, n. 2, p. 173–198, 2006. ISSN 0256-2499. Citation on page 28.

LEE, C.-H.; LIN, C.-R.; CHEN, M.-S. On mining general temporal association rules in a publication database. In: **Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on**. [S.l.: s.n.], 2001. p. 337–344. Citation on page 115.

LEE, D.-T. **Proximity and Reachability in the Plane**. Phd Thesis (PhD Thesis), Champaign, IL, USA, 1978. AAI7913526. Citation on page 95.

LESKOVEC, J.; LANG, K. J.; MAHONEY, M. Empirical comparison of algorithms for network community detection. In: **Proceedings of the 19th International Conference on World Wide Web**. New York, NY, USA: ACM, 2010. (WWW '10), p. 631–640. ISBN 978-1-60558-799-8. Citation on page 71.

LEWIS, D. D. Naive (bayes) at forty: The independence assumption in information retrieval. In: **Proceedings of the 10th European Conference on Machine Learning**. London, UK, UK: Springer-Verlag, 1998. (ECML '98), p. 4–15. ISBN 3-540-64417-2. Citation on page 50.

LI, C.; BISWAS, G.; DALE, M.; DALE, P. Building models of ecological dynamics using hmm based temporal data clustering — a preliminary study. In: HOFFMANN, F.; HAND, D.; ADAMS, N.; FISHER, D.; GUIMARAES, G. (Ed.). **Advances in Intelligent Data Analysis**. [S.l.]: Springer Berlin Heidelberg, 2001, (Lecture Notes in Computer Science, v. 2189). p. 53–62. ISBN 978-3-540-42581-6. Citation on page 27.

LI, C.-S.; YU, P. S.; CASTELLI, V. Malm: A framework for mining sequence database at multiple abstraction levels. In: **Proceedings of the Seventh International Conference on Information and Knowledge Management**. New York, NY, USA: ACM, 1998. (CIKM '98), p. 267–272. ISBN 1-58113-061-9. Citation on page 51.

LI, P.; WANG, B.-H. Extracting hidden fluctuation patterns of hang seng stock index from network topologies. **Physica A: Statistical Mechanics and its Applications**, v. 378, n. 2, p. 519 – 526, 2007. ISSN 0378-4371. Citation on page 74.

LI, X.; HAN, J.; KIM, S. Motion-alert: Automatic anomaly detection in massive moving objects. In: **Proceedings of the 4th IEEE International Conference on Intelligence and Security Informatics**. Berlin, Heidelberg: Springer-Verlag, 2006. (ISI'06), p. 166–177. ISBN 3-540-34478-0, 978-3-540-34478-0. Citation on page 54.

LI, X.; HAN, J.; KIM, S.; GONZALEZ, H. Roam: Rule- and motif-based anomaly detection in massive moving object data sets. In: **SDM**. [S.l.: s.n.], 2007. p. 273–284. Citation on page 54.

LI, X.-G.; GAO, Z.-Y.; LI, K.-P.; ZHAO, X.-M. Relationship between microscopic dynamics in traffic flow and complexity in networks. **Phys. Rev. E**, American Physical Society, v. 76, p. 016110, Jul 2007. Citation on page 74.

LIAN, X.; CHEN, L. Efficient similarity search over future stream time series. **Knowledge and Data Engineering, IEEE Transactions on**, v. 20, n. 1, p. 40–54, Jan 2008. ISSN 1041-4347. Citation on page 114.

LIAO, T. W. Clustering of time series data—a survey. **Pattern Recognition**, v. 38, n. 11, p. 1857 – 1874, 2005. ISSN 0031-3203. Citations on pages 35, 42, and 75.

LIEBSCHER, V. **gromovlab: Gromov-Hausdorff Type Distances for Labeled Metric Spaces**. [S.l.], 2015. R package version 0.7-6. Available: <<https://CRAN.R-project.org/package=gromovlab>>. Citation on page 96.

LIM, T.-S.; LOH, W.-Y.; SHIH, Y.-S. A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. **Mach. Learn.**, Kluwer Academic Publishers, Hingham, MA, USA, v. 40, n. 3, p. 203–228, Sep. 2000. ISSN 0885-6125. Citation on page 60.

LIN, J.; KEOGH, E.; LONARDI, S.; PATEL, P. Finding Motifs in Time Series. In: **Proceedings of the Second Workshop on Temporal Data Mining**. Edmonton, Alberta, Canada: [s.n.], 2002. Citation on page 57.

LIN, J.; KEOGH, E.; LONARDI, S.; LANKFORD, J. P.; NYSTROM, D. M. Visually mining and monitoring massive time series. In: **In Proceedings of the 10 th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. [S.l.]: ACM Press, 2004. p. 460–469. Citation on page 28.

LIN, J.; KEOGH, E.; WEI, L.; LONARDI, S. Experiencing sax: A novel symbolic representation of time series. **Data Min. Knowl. Discov.**, Kluwer Academic Publishers, Hingham, MA, USA, v. 15, n. 2, p. 107–144, Oct. 2007. ISSN 1384-5810. Citation on page 54.

LIN, J.-S.; KULIC, D. Online segmentation of human motion for automated rehabilitation exercise analysis. **Neural Systems and Rehabilitation Engineering, IEEE Transactions on**, v. 22, n. 1, p. 168–180, Jan 2014. ISSN 1534-4320. Citation on page 52.

LIN, T.-h.; KAMINSKI, N.; BAR-JOSEPH, Z. Alignment and classification of time series gene expression in clinical studies. **Bioinformatics**, v. 24, n. 13, p. i147–i155, 2008. Citation on page 27.

LIU, C. L. **Introduction to combinatorial mathematics / C.L. Liu**. [S.l.]: McGraw-Hill, New York :, 1968. ISBN 0070381240. Citation on page 41.

LORENZ, E. N. Atmospheric predictability as revealed by naturally occurring analogues. **Journal of the Atmospheric Sciences**, American Meteorological Society, v. 26, n. 4, p. 636–646, 2014/04/13 1969. Citation on page 59.

LUCAS, D. C. de. **Classification Techniques for Time Series and Functional Data**. Phd Thesis (PhD Thesis) — Universidad Carlos III de Madrid, 2003. Citation on page 80.

LUQUE, B.; LACASA, L.; BALLESTEROS, F.; LUQUE, J. Horizontal visibility graphs: Exact results for random time series. **Phys. Rev. E**, American Physical Society, v. 80, p. 046103, Oct 2009. Citations on pages 94 and 107.

MA, J.; PERKINS, S. Online novelty detection on temporal sequences. In: **Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: ACM, 2003. (KDD '03), p. 613–618. ISBN 1-58113-737-0. Citation on page 55.

_____. Time-series novelty detection using one-class support vector machines. In: **Neural Networks, 2003. Proceedings of the International Joint Conference on**. [S.l.: s.n.], 2003. v. 3, p. 1741–1745 vol.3. ISSN 1098-7576. Citation on page 54.

MACKAY, D. J. **Information theory, inference and learning algorithms**. [S.l.]: Cambridge university press, 2003. Citation on page 71.

MACQUEEN, J. B. Some methods of classification and analysis of multivariate observations. In: **Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability**. [S.l.: s.n.], 1967. p. 281–297. Citation on page 41.

MAHARAJ, E. Cluster of time series. **Journal of Classification**, Springer-Verlag, v. 17, n. 2, p. 297–314, 2000. ISSN 0176-4268. Citations on pages 85 and 87.

MALETZKE, A.; LEE, H.; ENRIQUE, G.; BATISTA, A.; COY, C.; FAGUNDES, J.; CHUNG, W. Time series classification with motifs and characteristics. In: ESPIN, R.; PÉREZ, R. B.; COBO, A.; MARX, J.; VALDÉS, A. R. (Ed.). **Soft Computing for Business Intelligence**. [S.l.]: Springer Berlin Heidelberg, 2014, (Studies in Computational Intelligence, v. 537). p. 125–138. ISBN 978-3-642-53736-3. Citation on page 57.

MARSLAND, S. **Machine Learning: An Algorithmic Perspective**. 1st. ed. [S.l.]: Chapman & Hall/CRC, 2009. ISBN 1420067184, 9781420067187. Citation on page 48.

MARWAN, N.; DONGES, J. F.; ZOU, Y.; DONNER, R. V.; KURTHS, J. Complex network approach for recurrence analysis of time series. **Physics Letters A**, v. 373, n. 46, p. 4246 – 4254, 2009. ISSN 0375-9601. Citations on pages 31, 73, and 94.

MASLOV, S.; SNEPPEN, K. Specificity and stability in topology of protein networks. **Science**, v. 296, n. 5569, p. 910–913, 2002. Citation on page 28.

MATEO, F.; CARRASCO, J. J.; SELLAMI, A.; MILLÁN-GIRALDO, M.; DOMÍNGUEZ, M.; SORIA-OLIVAS, E. Machine learning methods to forecast temperature in buildings. **Expert Syst. Appl.**, Pergamon Press, Inc., Tarrytown, NY, USA, v. 40, n. 4, p. 1061–1068, Mar. 2013. ISSN 0957-4174. Citation on page 61.

MEMOLI, F. On the use of Gromov-Hausdorff Distances for Shape Comparison. In: BOTSCH, M.; PAJAROLA, R.; CHEN, B.; ZWICKER, M. (Ed.). **Eurographics Symposium on Point-Based Graphics**. [S.l.]: The Eurographics Association, 2007. ISSN 1811-7813. Citation on page 96.

_____. Gromov-hausdorff distances in euclidean spaces. In: **2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops**. [S.l.: s.n.], 2008. p. 1–8. Citation on page 96.

MILO, R.; SHEN-ORR, S.; ITZKOVITZ, S.; KASHTAN, N.; CHKLOVSKII, D.; ALON, U. Network motifs: Simple building blocks of complex networks. **Science**, v. 298, n. 5594, p. 824–827, 2002. Citation on page 72.

MINNEN, D.; ISBELL, C.; ESSA, I.; STARNER, T. Detecting subdimensional motifs: An efficient algorithm for generalized multivariate pattern discovery. In: **Proceedings of the 2007 Seventh IEEE International Conference on Data Mining**. Washington, DC, USA: IEEE Computer Society, 2007. (ICDM '07), p. 601–606. ISBN 0-7695-3018-4. Citation on page 58.

MINNEN, D.; ISBELL, C. L.; ESSA, I.; STARNER, T. Discovering multivariate motifs using subsequence density estimation and greedy mixture learning. In: **Proceedings of the 22Nd National Conference on Artificial Intelligence - Volume 1**. [S.l.]: AAAI Press, 2007. (AAAI'07), p. 615–620. ISBN 978-1-57735-323-2. Citations on pages 57 and 58.

MITCHELL, M. **Complexity: A Guided Tour**. New York, NY, USA: Oxford University Press, Inc., 2009. Citation on page 63.

MITHAL, V.; O'CONNOR, Z.; STEINHAEUSER, K.; BORIAH, S.; KUMAR, V.; POTTER, C.; KLOOSTER, S. Time series change detection using segmentation: A case study for land cover monitoring. In: **Intelligent Data Understanding (CIDU), 2012 Conference on**. [S.l.: s.n.], 2012. p. 63–70. Citation on page 52.

MITSA, T. **Temporal Data Mining**. [S.l.]: Taylor & Francis, 2010. (Chapman & Hall/CRC Data Mining and Knowledge Discovery Series). ISBN 9781420089776. Citation on page 28.

MOHAMMAD, Y.; NISHIDA, T. Constrained motif discovery in time series. **New Generation Computing**, Verlag Omsha Tokio, v. 27, n. 4, p. 319–346, 2009. ISSN 0288-3635. Citation on page 58.

_____. Exact discovery of length-range motifs. In: NGUYEN, N.; ATTACHOO, B.; TRAWINSKI, B.; SOMBOONVIWAT, K. (Ed.). **Intelligent Information and Database Systems**. [S.l.]: Springer International Publishing, 2014, (Lecture Notes in Computer Science, v. 8398). p. 23–32. ISBN 978-3-319-05457-5. Citation on page 58.

MÖLLER-LEVET, C.; KLAWONN, F.; CHO, K.-H.; WOLKENHAUER, O. Fuzzy clustering of short time-series and unevenly distributed sampling points. In: BERTHOLD, M. R.; LENZ, H.-J.; BRADLEY, E.; KRUSE, R.; BORGELT, C. (Ed.). **Advances in Intelligent Data Analysis V**. [S.l.]: Springer Berlin Heidelberg, 2003, (Lecture Notes in Computer Science, v. 2810). p. 330–340. ISBN 978-3-540-40813-0. Citation on page 43.

_____. Fuzzy clustering of short time-series and unevenly distributed sampling points. In: **Advances in Intelligent Data Analysis V**. [S.l.]: Springer Berlin Heidelberg, 2003, (Lecture Notes in Computer Science, v. 2810). p. 330–340. ISBN 978-3-540-40813-0. Citation on page 80.

MORALES-ESTEBAN, A.; MARTÍNEZ-ÁLVAREZ, F.; TRONCOSO, A.; JUSTO, J. L.; RUBIO-ESCUADERO, C. Pattern recognition to forecast seismic time series. **Expert Syst. Appl.**, Pergamon Press, Inc., Tarrytown, NY, USA, v. 37, n. 12, p. 8333–8342, Dec. 2010. ISSN 0957-4174. Citation on page 27.

MÖRCHEN, F.; ULTSCH, A. Efficient mining of understandable patterns from multivariate interval time series. **Data Min. Knowl. Discov.**, Kluwer Academic Publishers, Hingham, MA, USA, v. 15, n. 2, p. 181–215, Oct. 2007. ISSN 1384-5810. Citation on page 115.

MORETTIN, P.; TOLOI, C. **Análise de séries temporais**. [S.l.]: Edgard Blucher, 2006. Citations on pages 27 and 36.

MORSE, M. D.; PATEL, J. M. An efficient and accurate method for evaluating time series similarity. In: **Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data**. New York, NY, USA: ACM, 2007. (SIGMOD '07), p. 569–580. ISBN 978-1-59593-686-8. Citation on page 38.

MUEEN, A. Enumeration of time series motifs of all lengths. In: **Data Mining (ICDM), 2013 IEEE 13th International Conference on**. [S.l.: s.n.], 2013. p. 547–556. ISSN 1550-4786. Citation on page 57.

_____. Time series motif discovery: dimensions and applications. **Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery**, John Wiley & Sons, Inc., v. 4, n. 2, p. 152–159, 2014. ISSN 1942-4795. Citation on page 58.

MUEEN, A.; KEOGH, E.; BIGDELY-SHAMLO, N. Finding Time Series Motifs in Disk-Resident Data. **2009 Ninth IEEE International Conference on Data Mining**, Ieee, p. 367–376, Dec. 2009. Citation on page 58.

MUEEN, A.; KEOGH, E. J.; ZHU, Q.; CASH, S.; WESTOVER, M. B. Exact discovery of time series motifs. In: **SDM**. [S.l.]: SIAM, 2009. p. 473–484. Citations on pages 57 and 58.

NAIRAC, A.; TOWNSEND, N.; CARR, R.; KING, S.; COWLEY, P.; TARASSENKO, L. A system for the analysis of jet engine vibration data. **Integr. Comput.-Aided Eng.**, IOS Press, Amsterdam, The Netherlands, The Netherlands, v. 6, n. 1, p. 53–66, Jan. 1999. ISSN 1069-2509. Citation on page 54.

NANOPOULOS, A.; ALCOCK, R.; MANOLOPOULOS, Y. Information processing and technology. In: MASTORAKIS, N.; NIKOLOPOULOS, S. D. (Ed.). Commack, NY, USA: Nova Science Publishers, Inc., 2001. chap. Feature-based Classification of Time-series Data, p. 49–61. ISBN 1-59033-116-8. Citation on page 49.

NARANG, A.; BHATTACHERJEE, S. Parallel exact time series motif discovery. In: D'AMBRA, P.; GUARRACINO, M.; TALIA, D. (Ed.). **Euro-Par 2010 - Parallel Processing**. [S.l.]: Springer Berlin Heidelberg, 2010, (Lecture Notes in Computer Science, v. 6272). p. 304–315. ISBN 978-3-642-15290-0. Citation on page 58.

NEAPOLITAN, R. E. **Learning Bayesian Networks**. [S.l.]: Prentice Hall, 2003. Citations on pages 27 and 59.

NEWMAN, M.; BARABASI, A.-L.; WATTS, D. J. **The Structure and Dynamics of Networks: (Princeton Studies in Complexity)**. Princeton, NJ, USA: Princeton University Press, 2006. ISBN 0691113572. Citations on pages 31 and 72.

NEWMAN, M.; GIRVAN, M. Finding and evaluating community structure in networks. **Physical Review E**, v. 69, n. 2, p. 026113, Feb. 2004. ISSN 1539-3755. Citation on page 70.

NEWMAN, M. E. J. The Structure and Function of Complex Networks. **SIAM Review**, v. 45, n. 2, p. 167–256, Jan. 2003. ISSN 0036-1445. Citation on page 28.

NEWMAN, M. E. J.; FORREST, S.; BALTHROP, J. Email networks and the spread of computer viruses. **Phys. Rev. E**, American Physical Society, v. 66, p. 035101, Sep 2002. Citation on page 28.

NG, R.; HAN, J. Clarans: a method for clustering objects for spatial data mining. **Knowledge and Data Engineering, IEEE Transactions on**, v. 14, n. 5, p. 1003–1016, Sep 2002. ISSN 1041-4347. Citation on page 41.

NICOLIS, G.; CANTÚ, A. G.; NICOLIS, C. Dynamical aspects of interaction networks. **International Journal of Bifurcation and Chaos**, v. 15, n. 11, p. 3467–3480, 2005. Citation on page 74.

_____. Dynamical aspects of interaction networks. **International Journal of Bifurcation and Chaos**, v. 15, n. 11, p. 3467–3480, 2005. Citation on page 94.

NUÑEZ, A.; LACASA, L.; VALERO, E.; GÓMEZ, J. P.; LUQUE, B. Detecting series periodicity with horizontal visibility graphs. **I. J. Bifurcation and Chaos**, v. 22, n. 7, 2012. Citation on page 106.

OLIVEIRA, T. de; ZHAO, L. Complex network community detection based on swarm aggregation. In: **Natural Computation, 2008. ICNC '08. Fourth International Conference on**. [S.l.: s.n.], 2008. v. 7, p. 604–608. Citation on page 30.

OLIVEIRA, T. de; ZHAO, L.; FACELI, K.; CARVALHO, A. C. P. L. F. D. Data clustering based on complex network community detection. In: **Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence). IEEE Congress on**. [S.l.: s.n.], 2008. p. 2121–2126. Citation on page 71.

OLIVER, J. J.; BAXTER, R. A.; WALLACE, C. S. Minimum message length segmentation. In: WU, X.; KOTAGIRI, R.; KORB, K. (Ed.). **Research and Development in Knowledge Discovery and Data Mining**. [S.l.: Springer Berlin Heidelberg, 1998, (Lecture Notes in Computer Science, v. 1394). p. 222–233. ISBN 978-3-540-64383-8. Citation on page 52.

OMRANIAN, N.; KLIE, S.; MUELLER-ROEBER, B.; NIKOLOSKE, Z. Network-based segmentation of biological multivariate time series. **PLoS ONE**, Public Library of Science, v. 8, n. 5, p. e62974, 05 2013. Citation on page 52.

ORMAN, G.; LABATUT, V.; CHERIFI, H. Qualitative comparison of community detection algorithms. In: CHERIFI, H.; ZAIN, J.; EL-QAWASMEH, E. (Ed.). **Digital Information and Communication Technology and Its Applications**. [S.l.: Springer Berlin Heidelberg, 2011, (Communications in Computer and Information Science, v. 167). p. 265–279. ISBN 978-3-642-22026-5. Citation on page 71.

OUYANG, R.; REN, L.; CHENG, W.; ZHOU, C. Similarity search and pattern discovery in hydrological time series data mining. **Hydrological Processes**, John Wiley & Sons, Ltd., v. 24, n. 9, p. 1198–1210, 2010. ISSN 1099-1085. Citation on page 27.

PALLA, G.; DERENYI, I.; FARKAS, I.; VICSEK, T. Uncovering the overlapping community structure of complex networks in nature and society. **Nature**, v. 435, n. 7043, p. 814–818, 06 2005. Citation on page 71.

PAPADIMITRIOU, P.; DASDAN, A.; GARCIA-MOLINA, H. Web graph similarity for anomaly detection. **Journal of Internet Services and Applications**, Springer-Verlag, v. 1, n. 1, p. 19–30, 2010. ISSN 1867-4828. Citation on page 96.

PAPADIMITRIOU, S.; BROCKWELL, A.; FALOUTSOS, C. Adaptive, hands-off stream mining. In: **Proceedings of the 29th International Conference on Very Large Data Bases - Volume 29**. [S.l.]: VLDB Endowment, 2003. (VLDB '03), p. 560–571. ISBN 0-12-722442-4. Citation on page 106.

PARK, S.; KIM, S.-W.; CHU, W. W. Segment-based approach for subsequence searches in sequence databases. In: **Proceedings of the 2001 ACM Symposium on Applied Computing**. New York, NY, USA: ACM, 2001. (SAC '01), p. 248–252. ISBN 1-58113-287-5. Citation on page 51.

PARK, S.; LEE, D.; CHU, W. W. Fast retrieval of similar subsequences in long sequence databases. In: **Proceedings of the 1999 Workshop on Knowledge and Data Engineering Exchange**. Washington, DC, USA: IEEE Computer Society, 1999. (KDEX '99), p. 60. ISBN 0-7695-0453-1. Citation on page 51.

PARTHASARATHY, S.; MEHTA, S.; SRINIVASAN, S. Robust periodicity detection algorithms. In: **Proceedings of the 15th ACM International Conference on Information and Knowledge Management**. New York, NY, USA: ACM, 2006. (CIKM '06), p. 874–875. ISBN 1-59593-433-2. Citation on page 105.

PAVLOVIC, V.; FREY, B.; HUANG, T. Time-series classification using mixed-state dynamic bayesian networks. In: **Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on**. [S.l.: s.n.], 1999. v. 2, p. 615 Vol. 2. ISSN 1063-6919. Citation on page 50.

PERCIVAL, D. B.; WALDEN, A. T. **Wavelet Methods for Time Series Analysis**. [S.l.]: Cambridge University Press, 2000. (Cambridge Series in Statistical and Probabilistic Mathematics). Citation on page 27.

PINCOMBE, B. Anomaly Detection in Time Series of Graphs using ARMA Processes. **ASOR Bulletin**, v. 24, n. 4, p. 2–10, 2005. Citation on page 55.

PONS, P.; LATAPY, M. Computing communities in large networks using random walks. In: YOLUM, p.; GÜNGÖR, T.; GÜRGEN, F.; ÖZTURAN, C. (Ed.). **Computer and Information Sciences - ISCIS 2005**. [S.l.]: Springer Berlin Heidelberg, 2005, (Lecture Notes in Computer Science, v. 3733). p. 284–293. ISBN 978-3-540-29414-6. Citations on pages 30 and 81.

PORTER, M. A.; ONNELA, J.-P.; MUCHA, P. J. Communities in networks. **Notices of the American Mathematical Society**, v. 56, n. 9, p. 1082–1097, 2009. Citations on pages 30 and 72.

PORTNOY, L.; ESKIN, E.; STOLFO, S. Intrusion detection with unlabeled data using clustering. In: **In Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001)**. [S.l.: s.n.], 2001. p. 5–8. Citation on page 54.

POVINELLI, R.; JOHNSON, M.; LINDGREN, A.; YE, J. Time series classification using gaussian mixture models of reconstructed phase spaces. **Knowledge and Data Engineering, IEEE Transactions on**, v. 16, n. 6, p. 779–783, June 2004. ISSN 1041-4347. Citation on page 50.

PREKOPCSÁK, Z.; LEMIRE, D. Time series classification by class-specific mahalanobis distance measures. **Advances in Data Analysis and Classification**, Springer-Verlag, v. 6, n. 3, p. 185–200, 2012. ISSN 1862-5347. Citation on page 49.

PYLE, D. **Data Preparation for Data Mining**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999. ISBN 1-55860-529-0. Citation on page 37.

QU, Y.; WANG, C.; WANG, X. S. Supporting fast search in time series for movement patterns in multiple scales. In: **Proceedings of the Seventh International Conference on Information and Knowledge Management**. New York, NY, USA: ACM, 1998. (CIKM '98), p. 251–258. ISBN 1-58113-061-9. Citation on page 51.

QUILES, M. G.; ZHAO, L.; ALONSO, R. L.; ROMERO, R. A. F. Particle competition for complex network community detection. **Chaos: An Interdisciplinary Journal of Nonlinear Science**, v. 18, n. 3, p. 033107–033107, 2008. Citations on pages 30 and 70.

QUINLAN, J. R. Induction of decision trees. **Mach. Learn.**, Kluwer Academic Publishers, Hingham, MA, USA, v. 1, n. 1, p. 81–106, Mar. 1986. ISSN 0885-6125. Citation on page 46.

_____. **C4.5: Programs for Machine Learning**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993. ISBN 1-55860-238-0. Citation on page 46.

RADICCHI, F.; CASTELLANO, C.; CECCONI, F.; LORETO, V.; PARISI, D. Defining and identifying communities in networks. **Proceedings of the National Academy of Sciences of the United States of America**, v. 101, n. 9, p. 2658–2663, 2004. Citation on page 70.

RAGHAVAN, U. N.; ALBERT, R.; KUMARA, S. Near linear time algorithm to detect community structures in large-scale networks. **Phys. Rev. E**, American Physical Society, v. 76, p. 036106, Sep 2007. Citations on pages 30, 71, and 81.

RAMONI, M.; SEBASTIANI, P.; COHEN, P. Bayesian clustering by dynamics. **Mach. Learn.**, Kluwer Academic Publishers, Hingham, MA, USA, v. 47, n. 1, p. 91–121, Apr. 2002. ISSN 0885-6125. Citation on page 44.

RAMONI, M.; SEBASTIANI, P.; COHEN, P. R. Multivariate clustering by dynamics. In: KAUTZ, H. A.; PORTER, B. W. (Ed.). **AAAI/IAAI**. [S.l.]: AAAI Press / The MIT Press, 2000. p. 633–638. ISBN 0-262-51112-6. Citation on page 27.

RANI, S.; SIKKA, G. Recent techniques of clustering of time series data: A survey. **International Journal of Computer Applications**, v. 52, n. 15, p. 1–9, August 2012. Published by Foundation of Computer Science, New York, USA. Citation on page 42.

RAPOPORT, A. Spread of information through a population with socio-structural bias: I. assumption of transitivity. **The bulletin of mathematical biophysics**, Kluwer Academic Publishers, v. 15, n. 4, p. 523–533, 1953. ISSN 0007-4985. Citation on page 66.

RASHEED, F.; ALSHALALFA, M.; ALHAJJ, R. Efficient periodicity mining in time series databases using suffix trees. **Knowledge and Data Engineering, IEEE Transactions on**, v. 23, n. 1, p. 79–94, Jan 2011. ISSN 1041-4347. Citation on page 106.

RASMUSSEN, C. E.; WILLIAMS, C. K. I. **Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)**. [S.l.]: The MIT Press, 2005. ISBN 026218253X. Citation on page 59.

RASOULI, K.; HSIEH, W. W.; CANNON, A. J. Daily streamflow forecasting by machine learning methods with weather and climate inputs. **Journal of Hydrology**, v. 414–415, n. 0, p. 284 – 293, 2012. ISSN 0022-1694. Citation on page 61.

RATANAMAHATANA, C. A.; KEOGH, E. J. Making time-series classification more accurate using learned constraints. In: BERRY, M. W.; DAYAL, U.; KAMATH, C.; SKILLICORN, D. B. (Ed.). **SDM**. [S.l.]: SIAM, 2004. p. 11–22. ISBN 0-89871-568-7. Citations on pages 38 and 49.

RATTIGAN, M. J.; MAIER, M.; JENSEN, D. Graph clustering with network structure indices. In: **Proceedings of the 24th International Conference on Machine Learning**. New York, NY, USA: ACM, 2007. (ICML '07), p. 783–790. ISBN 978-1-59593-793-3. Citation on page 70.

REICHARDT, J.; WHITE, D. R. Role models for complex networks. **The European Physical Journal B**, v. 60, n. 2, p. 217–224, 2007. ISSN 1434-6036. Citation on page 71.

REYES, J.; MORALES-ESTEBAN, A.; MARTÍNEZ-ÁLVAREZ, F. Neural networks to predict earthquakes in chile. **Applied Soft Computing**, v. 13, n. 2, p. 1314 – 1328, 2013. ISSN 1568-4946. Citation on page 61.

RISH, I. An empirical study of the naive bayes classifier. **IJCAI 2001 workshop on empirical methods in artificial intelligence**, v. 3, n. 22, p. 41–46, 2001. Citation on page 47.

RODRIGUES, P.; GAMA, J.; PEDROSO, J. Hierarchical clustering of time-series data streams. **Knowledge and Data Engineering, IEEE Transactions on**, v. 20, n. 5, p. 615–627, May 2008. ISSN 1041-4347. Citation on page 43.

ROMILLY, P. Time series modelling of global mean temperature for managerial decision-making. **Journal of Environmental Management**, v. 76, n. 1, p. 61 – 70, 2005. ISSN 0301-4797. Citation on page 27.

ROSENBLATT, F. **Principles of neurodynamics: perceptrons and the theory of brain mechanisms**. [S.l.]: Spartan Books, 1962. (Report (Cornell Aeronautical Laboratory)). Citation on page 46.

ROSVALL, M.; BERGSTROM, C. T. An information-theoretic framework for resolving community structure in complex networks. **Proceedings of the National Academy of Sciences**, v. 104, n. 18, p. 7327–7331, 2007. Citation on page 71.

_____. Maps of random walks on complex networks reveal community structure. **Proceedings of the National Academy of Sciences**, v. 105, n. 4, p. 1118–1123, 2008. Citations on pages 30 and 81.

ROUSSEEUW, P. J.; LEROY, A. M. **Robust Regression and Outlier Detection**. New York, NY, USA: John Wiley & Sons, Inc., 1987. ISBN 0-471-85233-3. Citation on page 52.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. In: RUMELHART, D. E.; MCCLELLAND, J. L.; GROUP, C. P. R. (Ed.). Cambridge, MA, USA: MIT Press, 1986. chap. Learning Internal Representations by Error Propagation, p. 318–362. ISBN 0-262-68053-X. Citation on page 47.

SAFAVIAN, S.; LANDGREBE, D. A survey of decision tree classifier methodology. **Systems, Man and Cybernetics, IEEE Transactions on**, v. 21, n. 3, p. 660–674, May 1991. ISSN 0018-9472. Citation on page 46.

SALVADOR, S.; CHAN, P. Learning states and rules for detecting anomalies in time series. **Applied Intelligence**, Kluwer Academic Publishers, Hingham, MA, USA, v. 23, n. 3, p. 241–255, Dec. 2005. ISSN 0924-669X. Citation on page 43.

SAMÉ, A.; CHAMROUKHI, F.; GOVAERT, G.; AKNIN, P. Model-based clustering and segmentation of time series with changes in regime. **Advances in Data Analysis and Classification**, Springer-Verlag, v. 5, n. 4, p. 301–321, 2011. ISSN 1862-5347. Citation on page 45.

SAPANKEVYCH, N. I.; SANKAR, R. Time series prediction using support vector machines: A survey. **Comp. Intell. Mag.**, IEEE Press, Piscataway, NJ, USA, v. 4, n. 2, p. 24–38, May 2009. ISSN 1556-603X. Citation on page 58.

SARKER, B. K.; MORI, T.; HIRATA, T.; UEHARA, K. Parallel algorithms for mining association rules in time series data. In: **Proceedings of the 2003 International Conference on Parallel and Distributed Processing and Applications**. Berlin, Heidelberg: Springer-Verlag, 2003. (ISPA'03), p. 273–284. ISBN 3-540-40523-2. Citation on page 115.

SAYAMA, H.; PESTOV, I.; SCHMIDT, J.; BUSH, B. J.; WONG, C.; YAMANOI, J.; GROSS, T. Modeling complex systems with adaptive networks. **Computers & Mathematics with Applications**, Elsevier Ltd, v. 65, n. 10, p. 1645–1664, May 2013. ISSN 08981221. Citation on page 28.

SCHITTENKOPF, C.; TIÑO, P.; DORFFNER, G. The benefit of information reduction for trading strategies. **Applied Economics**, v. 34, n. 7, p. 917–930, 2002. Citation on page 44.

SCOTT, J. **Social Network Analysis: A Handbook**. [S.l.]: SAGE Publications, 2000. ISBN 9780761963394. Citation on page 28.

SEKER, S.; MERT, C.; AL-NAAMI, K.; AYAN, U.; OZALP, N. Ensemble classification over stock market time series and economy news. In: **Intelligence and Security Informatics (ISI), 2013 IEEE International Conference on**. [S.l.: s.n.], 2013. p. 272–273. Citation on page 50.

SEQUEIRA, K.; ZAKI, M. Admit: Anomaly-based data mining for intrusions. In: **Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: ACM, 2002. (KDD '02), p. 386–395. ISBN 1-58113-567-X. Citation on page 56.

SERRÀ, J.; ARCOS, J. L. **An Empirical Evaluation of Similarity Measures for Time Series Classification**. 2014. <http://arxiv.org/abs/1401.3973v1>. Citation on page 38.

SHAO, Z.-G. Network analysis of human heartbeat dynamics. **Applied Physics Letters**, v. 96, n. 7, p. 073703–073703–3, Feb 2010. ISSN 0003-6951. Citation on page 74.

SHATKAY, H.; ZDONIK, S. Approximate queries and representations for large data sequences. In: **Data Engineering, 1996. Proceedings of the Twelfth International Conference on**. [S.l.: s.n.], 1996. p. 536–545. ISSN 1063-6382. Citation on page 51.

SHIEH, J.; KEOGH, E. iSAX: Indexing and Mining Terabyte Sized Time Series. In: **Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: ACM, 2008. (KDD '08), p. 623–631. ISBN 978-1-60558-193-4. Citation on page 114.

SHIMADA, Y.; KIMURA, T.; IKEGUCHI, T. Analysis of chaotic dynamics using measures of the complex network theory. In: KŮRKOVÁ, V.; NERUDA, R.; KOUTNÍK, J. (Ed.). **Artificial Neural Networks - ICANN 2008**. [S.l.]: Springer Berlin Heidelberg, 2008, (Lecture Notes in Computer Science, v. 5163). p. 61–70. ISBN 978-3-540-87535-2. Citation on page 73.

SHIRAZI, A. H.; JAFARI, G. R.; DAVOUDI, J.; PEINKE, J.; TABAR, M. R. R.; SAHIMI, M. Mapping stochastic processes onto complex networks. **Journal of Statistical Mechanics: Theory and Experiment**, v. 2009, n. 07, p. P07046, 2009. Citation on page 74.

SHUMWAY, R. H.; STOFFER, D. S. **Time Series Analysis and Its Applications: With R Examples**. [S.l.]: Springer, 2011. Citations on pages 27 and 60.

SI, Y.-W.; YIN, J. Obst-based segmentation approach to financial time series. **Engineering Applications of Artificial Intelligence**, v. 26, n. 10, p. 2581 – 2596, 2013. ISSN 0952-1976. Citation on page 52.

SILVA, D.; SOUZA, V. D.; BATISTA, G. Time series classification using compression distance of recurrence plots. In: **Data Mining (ICDM), 2013 IEEE 13th International Conference on**. [S.l.: s.n.], 2013. p. 687–696. ISSN 1550-4786. Citation on page 49.

SILVA, T. C.; ZHAO, L. Stochastic competitive learning in complex networks. **IEEE Trans. Neural Networks and Learning Systems**, v. 23, p. 385–397, 2012. Citations on pages 76 and 78.

SILVESTRI, G.; VERONA, F.; INNOCENTI, M.; NAPOLITANO, M. Fault detection using neural networks. In: **Neural Networks, 1994. IEEE World Congress on Computational Intelligence., 1994 IEEE International Conference on**. [S.l.: s.n.], 1994. v. 6, p. 3796–3799 vol.6. Citation on page 55.

SIMON, G.; LEE, J. A.; VERLEYSEN, M. Unfolding preprocessing for meaningful time series clustering. **Neural Networks**, v. 19, n. 6-7, p. 877–888, 2006. Citation on page 45.

SMALL, M. Complex networks from time series: Capturing dynamics. In: **Circuits and Systems (ISCAS), 2013 IEEE International Symposium on**. [S.l.: s.n.], 2013. p. 2509–2512. ISSN 0271-4302. Citation on page 31.

SMALL, M.; ZHANG, J.; XU, X. Transforming time series into complex networks. In: ZHOU, J. (Ed.). **Complex Sciences**. [S.l.]: Springer Berlin Heidelberg, 2009, (Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, v. 5). p. 2078–2089. ISBN 978-3-642-02468-9. Citation on page 73.

SNEATH, P. H. A. The application of computers to taxonomy. **Journal of general microbiology**, v. 17, n. 1, p. 201–226, Aug. 1957. ISSN 0022-1287. Citation on page 40.

SOKAL, R. R.; MICHENER, C. D. A statistical method for evaluating systematic relationships. **University of Kansas Scientific Bulletin**, v. 28, p. 1409–1438, 1958. Citation on page 40.

SONG, H.; LI, G. Tourism demand modelling and forecasting—a review of recent research. **Tourism Management**, v. 29, n. 2, p. 203 – 220, 2008. ISSN 0261-5177. Citation on page 27.

SØRENSEN, T. A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on Danish commons. **Biol. Skr.**, v. 5, p. 1–34, 1948. Citation on page 40.

SORJAMAA, A.; HAO, J.; REYHANI, N.; JI, Y.; LENDASSE, A. Methodology for long-term prediction of time series. **Neurocomputing**, v. 70, n. 16–18, p. 2861 – 2869, 2007. ISSN 0925-2312. Citation on page 59.

SOUNDARAJAN, S.; ELIASSI-RAD, T.; GALLAGHER, B. A guide to selecting a network similarity method. In: **Proceedings of the SIAM International Conference on Data Mining (SDM)**. [S.l.: s.n.], 2014. p. 1037–1045. Citation on page 96.

SPIRIN, V.; MIRNY, L. A. Protein complexes and functional modules in molecular networks. **Proceedings of the National Academy of Sciences**, v. 100, n. 21, p. 12123–12128, 2003. Citation on page 69.

SPORNS, O. Network analysis, complexity, and brain function. **Complexity**, Wiley Subscription Services, Inc., A Wiley Company, v. 8, n. 1, p. 56–60, 2002. ISSN 1099-0526. Citation on page 28.

STEINBACH, M.; TAN, P.-N.; KUMAR, V.; KLOOSTER, S.; POTTER, C. Discovery of climate indices using clustering. In: **Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: ACM, 2003. (KDD '03), p. 446–455. ISBN 1-58113-737-0. Citation on page 45.

STROGATZ, S. H. Exploring complex networks. **Nature**, v. 410, n. 6825, p. 268–76, Mar. 2001. ISSN 0028-0836. Citations on pages 28 and 65.

SUGIMURA, H.; MATSUMOTO, K. Classification system for time series data based on feature pattern extraction. In: **Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on**. [S.l.: s.n.], 2011. p. 1340–1345. ISSN 1062-922X. Citation on page 49.

TAN, P.-N.; STEINBACH, M.; KUMAR, V. **Introduction to Data Mining**. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2005. ISBN 0321321367. Citations on pages 32, 39, 41, and 48.

TANAKA, Y.; IWAMOTO, K.; UEHARA, K. Discovery of time-series motif from multi-dimensional data based on mdl principle. **Mach. Learn.**, Kluwer Academic Publishers, Hingham, MA, USA, v. 58, n. 2-3, p. 269–300, Feb. 2005. ISSN 0885-6125. Citations on pages 57 and 58.

TANG, H.; LIAO, S. S. Discovering original motifs with different lengths from time series. **Know.-Based Syst.**, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 21, n. 7, p. 666–671, Oct. 2008. ISSN 0950-7051. Citation on page 57.

TRAN, D.; WAGNER, M. Fuzzy c-means clustering-based speaker verification. In: **Proceedings of the 2002 AFSS International Conference on Fuzzy Systems. Calcutta: Advances in Soft Computing**. London, UK, UK: Springer-Verlag, 2002. (AFSS '02), p. 318–324. ISBN 3-540-43150-0. Citation on page 27.

TRAUD, A.; KELSIC, E.; MUCHA, P.; PORTER, M. Comparing community structure to characteristics in online collegiate social networks. **SIAM Review**, v. 53, n. 3, p. 526–543, 2011. Citations on pages 30 and 72.

TYLER, J. R.; WILKINSON, D. M.; HUBERMAN, B. A. Communities and technologies. In: HUYSMAN, M.; WENGER, E.; WULF, V. (Ed.). Deventer, The Netherlands, The Netherlands: Kluwer, B.V., 2003. chap. Email As Spectroscopy: Automated Discovery of Community Structure Within Organizations, p. 81–96. ISBN 1-4020-1611-5. Citation on page 30.

VESPIER, U.; NIJSSEN, S.; KNOBBE, A. Mining characteristic multi-scale motifs in sensor-based time series. In: **Proceedings of the 22Nd ACM International Conference on Conference on Information & Knowledge Management**. New York, NY, USA: ACM, 2013. (CIKM '13), p. 2393–2398. ISBN 978-1-4503-2263-8. Citation on page 58.

VLACHOS, M.; KOLLIOS, G.; GUNOPULOS, D. Discovering similar multidimensional trajectories. In: **Data Engineering, 2002. Proceedings. 18th International Conference on**. [S.l.: s.n.], 2002. p. 673–684. ISSN 1063-6382. Citation on page 38.

VLACHOS, M.; LIN, J.; KEOGH, E.; GUNOPULOS, D. A wavelet-based anytime algorithm for k-means clustering of time series. In: **In Proc. Workshop on Clustering High Dimensionality Data and Its Applications**. [S.l.: s.n.], 2003. p. 23–30. Citations on pages 27 and 44.

VLACHOS, M.; YU, P. S.; CASTELLI, V. On periodicity detection and structural periodic similarity. In: **SDM**. [S.l.: s.n.], 2005. p. 449–460. Citation on page 105.

VLACHOS, M.; YU, P. S.; CASTELLI, V.; MEEK, C. Structural periodic measures for time-series data. **Data Min. Knowl. Discov.**, Kluwer Academic Publishers, Hingham, MA, USA, v. 12, n. 1, p. 1–28, Jan. 2006. ISSN 1384-5810. Citation on page 106.

WANG, C.; WANG, X. S. Supporting content-based searches on time series via approximation. In: **Scientific and Statistical Database Management, 2000. Proceedings. 12th International Conference on**. [S.l.: s.n.], 2000. p. 69–81. ISSN 1099-3371. Citation on page 51.

WANG, W.; YANG, J.; MUNTZ, R. R. Sting: A statistical information grid approach to spatial data mining. In: **Proceedings of the 23rd International Conference on Very Large Data Bases**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997. (VLDB '97), p. 186–195. ISBN 1-55860-470-7. Citation on page 41.

WANG, X.; MUEEN, A.; DING, H.; TRAJCEVSKI, G.; SCHEUERMANN, P.; KEOGH, E. Experimental comparison of representation methods and distance measures for time series data. **Data Mining and Knowledge Discovery**, Springer US, v. 26, n. 2, p. 275–309, 2013. ISSN 1384-5810. Citations on pages 31, 75, 77, 93, and 98.

WASSERMAN, S.; FAUST, K.; IACOBUCCI, D. **Social Network Analysis : Methods and Applications (Structural Analysis in the Social Sciences)**. [S.l.]: Cambridge University Press, 1994. Paperback. Citation on page 28.

WATTS, D. J.; STROGATZ, S. H. Collective dynamics of 'small-world' networks. **Nature**, v. 393, n. 6684, p. 440–442, 06 1998. Citations on pages 28, 29, and 66.

WEI, L.; KEOGH, E. Semi-supervised time series classification. In: **Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: ACM, 2006. (KDD '06), p. 748–753. ISBN 1-59593-339-5. Citations on pages 49 and 50.

WEI, L.; KEOGH, E.; XI, X. Sexually explicit images: Finding unusual shapes. In: **Proceedings of the Sixth International Conference on Data Mining**. Washington, DC, USA: IEEE Computer Society, 2006. (ICDM '06), p. 711–720. ISBN 0-7695-2701-9. Citation on page 55.

WEIGEND, A.; GERSHENFELD, N. **Time Series Prediction: Forecasting the Future and Understanding the Past : Proceedings of the NATO Advanced Research Workshop on Comparative Time Series Analysis, Held in Santa Fe, New Mexico, May 14-17, 1992**. [S.l.]: Addison-Wesley, 1994. (Advanced book program). ISBN 9780201626025. Citation on page 28.

WEIHS, C.; LIGGES, U.; SOMMER, K. Analysis of music time series. In: RIZZI, A.; VICHI, M. (Ed.). **Compstat 2006 - Proceedings in Computational Statistics**. [S.l.]: Physica-Verlag HD, 2006. p. 147–159. ISBN 978-3-7908-1708-9. Citation on page 27.

WEISS, G. M. Mining with rarity: A unifying framework. **SIGKDD Explor. Newsl.**, ACM, New York, NY, USA, v. 6, n. 1, p. 7–19, Jun. 2004. ISSN 1931-0145. Citation on page 28.

WESTON, J.; LESLIE, C.; IE, E.; ZHOU, D.; ELISSEEFF, A.; NOBLE, W. S. Semi-supervised protein classification using cluster kernels. **Bioinformatics**, v. 21, n. 15, p. 3241–3247, 2005. Citation on page 50.

WIJK, J. V.; SELOW, E. V. Cluster and calendar based visualization of time series data. In: **Information Visualization, 1999. (Info Vis '99) Proceedings. 1999 IEEE Symposium on**. [S.l.: s.n.], 1999. p. 4–9, 140. ISSN 1522-404X. Citation on page 43.

WILKINSON, D. M.; HUBERMAN, B. A. A method for finding communities of related genes. **Proc. National Academy of Sciences U.S.A.**, v. 10, n. 1073, 2004. Citations on pages 70 and 71.

WINDISCHBERGER, C.; BARTH, M.; LAMM, C.; SCHROEDER, L.; BAUER, H.; GUR, R. C.; MOSER, E. Fuzzy cluster analysis of high-field functional mri data. **Artif. Intell. Med.**, Elsevier Science Publishers Ltd., Essex, UK, v. 29, n. 3, p. 203–223, Nov. 2003. ISSN 0933-3657. Citation on page 43.

WITTEN, I.; FRANK, E.; HALL, M. **Data Mining: Practical Machine Learning Tools and Techniques: Practical Machine Learning Tools and Techniques**. [S.l.]: Elsevier Science, 2011. (The Morgan Kaufmann Series in Data Management Systems). ISBN 9780080890364. Citation on page 39.

XI, X.; KEOGH, E.; SHELTON, C.; WEI, L.; RATANAMAHATANA, C. A. Fast time series classification using numerosity reduction. In: **Proceedings of the 23rd International Conference on Machine Learning**. New York, NY, USA: ACM, 2006. (ICML '06), p. 1033–1040. ISBN 1-59593-383-2. Citations on pages 32 and 49.

XING, Z.; PEI, J.; KEOGH, E. A brief survey on sequence classification. **ACM SIGKDD Explorations Newsletter**, v. 12, n. 1, p. 40, Nov. 2010. ISSN 19310145. Citations on pages 49, 93, and 98.

XIONG, Y.; YEUNG, D.-Y. Mixtures of arma models for model-based time series clustering. In: **Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on**. [S.l.: s.n.], 2002. p. 717–720. Citation on page 44.

XU, K.; JIANG, Y.; TANG, M.; YUAN, C.; TANG, C. Presee: An mdl/mml algorithm to time-series stream segmenting. **The Scientific World Journal**, v. 2013, p. 11, 2013. Citation on page 52.

XU, R.; WUNSCH, D. Survey of clustering algorithms. **Neural Networks, IEEE Transactions on**, IEEE, v. 16, n. 3, p. 645–678, May 2005. ISSN 1045-9227. Citation on page 40.

_____. **Clustering**. [S.l.]: Wiley, 2008. (IEEE Press Series on Computational Intelligence). ISBN 9780470382783. Citations on pages 42 and 94.

XU, X.; ZHANG, J.; SMALL, M. Superfamily phenomena and motifs of networks induced from time series. **Proceedings of the National Academy of Sciences**, v. 105, n. 50, p. 19601–19605, 2008. Citation on page 73.

YAIRI, T.; KATO, Y.; HORI, K. Fault detection by mining association rules from house-keeping data. In: **In Proc. of International Symposium on Artificial Intelligence, Robotics and Automation in Space**. [S.l.: s.n.], 2001. Citation on page 44.

YANG, B.; LIU, J. Discovering global network communities based on local centralities. **ACM Trans. Web**, ACM, New York, NY, USA, v. 2, n. 1, p. 9:1–9:32, Mar. 2008. ISSN 1559-1131. Citation on page 70.

YANG, Q.; WU, X. 10 Challenging Problems in Data Mining Research. **International Journal of Information Technology & Decision Making**, v. 05, n. 04, p. 597–604, Dec. 2006. ISSN 0219-6220. Citation on page 32.

YANG, Y.; YANG, H. Complex network-based time series analysis. **Physica A: Statistical Mechanics and its Applications**, v. 387, n. 5–6, p. 1381 – 1386, 2008. ISSN 0378-4371. Citations on pages 28, 31, and 73.

YANG, Y.; YANG, Q.; LU, W.; PAN, J.; PAN, R.; LU, C.; LI, L.; QIN, Z. Preprocessing time series data for classification with application to crm. In: **Proceedings of the 18th Australian Joint Conference on Advances in Artificial Intelligence**. Berlin, Heidelberg: Springer-Verlag, 2005. (AI'05), p. 133–142. ISBN 3-540-30462-2, 978-3-540-30462-3. Citation on page 50.

YANKOV, D.; KEOGH, E.; MEDINA, J.; CHIU, B.; ZORDAN, V. Detecting time series motifs under uniform scaling. In: **Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: ACM, 2007. (KDD '07), p. 844–853. ISBN 978-1-59593-609-7. Citations on pages 57 and 58.

YANKOV, D.; KEOGH, E.; REBBAPRAGADA, U. Disk aware discord discovery: Finding unusual time series in terabyte sized datasets. In: **Proceedings of the 2007 Seventh IEEE International Conference on Data Mining**. Washington, DC, USA: IEEE Computer Society, 2007. (ICDM '07), p. 381–390. ISBN 0-7695-3018-4. Citation on page 56.

YE, L.; KEOGH, E. Time series shapelets: A new primitive for data mining. In: **Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: ACM, 2009. (KDD '09), p. 947–956. ISBN 978-1-60558-495-9. Citations on pages 49 and 98.

YI, B.-K.; FALOUTSOS, C. Fast time sequence indexing for arbitrary lp norms. In: **Proceedings of the 26th International Conference on Very Large Data Bases**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000. (VLDB '00), p. 385–394. ISBN 1-55860-715-3. Citation on page 80.

ZAKI, M. J.; JR, W. M. **Data Mining and Analysis: Fundamental Concepts and Algorithms**. New York, NY, USA: Cambridge University Press, 2014. Citation on page 35.

ZHANG, G. Neural networks for classification: a survey. **Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on**, v. 30, n. 4, p. 451–462, Nov 2000. ISSN 1094-6977. Citation on page 47.

ZHANG, G.; PATUWO, B. E.; HU, M. Y. Forecasting with artificial neural networks:: The state of the art. **International Journal of Forecasting**, v. 14, n. 1, p. 35–62, Mar. 1998. Citation on page 58.

ZHANG, J.; SMALL, M. Complex network from pseudoperiodic time series: Topology versus dynamics. **Phys. Rev. Lett.**, American Physical Society, v. 96, p. 238701, Jun 2006. Citations on pages 31, 72, 80, and 94.

ZHANG, J.; SUN, J.; LUO, X.; ZHANG, K.; NAKAMURA, T.; SMALL, M. Characterizing pseudoperiodic time series through the complex network approach. **Physica D: Nonlinear Phenomena**, v. 237, n. 22, p. 2856 – 2865, 2008. ISSN 0167-2789. Citation on page 72.

ZHANG, T.; RAMAKRISHNAN, R.; LIVNY, M. Birch: An efficient data clustering method for very large databases. In: **Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data**. New York, NY, USA: ACM, 1996. (SIGMOD '96), p. 103–114. ISBN 0-89791-794-4. Citation on page 40.

ZHANG, X.; LIU, J.; DU, Y.; LV, T. A novel clustering method on time series data. **Expert Syst. Appl.**, Pergamon Press, Inc., Tarrytown, NY, USA, v. 38, n. 9, p. 11891–11900, Sep. 2011. ISSN 0957-4174. Citations on pages 85 and 87.

ZHAO, L.; CUPERTINO, T. H.; JR., J. a. R. B. Chaotic synchronization in general network topology for scene segmentation. **Neurocomputing**, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 71, n. 16-18, p. 3360–3366, Oct. 2008. ISSN 0925-2312. Citations on pages 30 and 71.

ZHONG, S. Semi-supervised sequence classification with hmms. In: BARR, V.; MARKOV, Z. (Ed.). **FLAIRS Conference**. [S.l.]: AAAI Press, 2004. p. 568–574. ISBN 1-57735-201-7. Citation on page 50.

ZHOU, H. Distance, dissimilarity index, and network community structure. **Phys. Rev. E**, American Physical Society, v. 67, p. 061901, Jun 2003. Citation on page 70.

ZIV, E.; MIDDENDORF, M.; WIGGINS, C. H. Information-theoretic approach to network modularity. **Phys. Rev. E**, American Physical Society, v. 71, p. 046117, Apr 2005. Citation on page 71.

ZOU, Y.; SMALL, M.; LIU, Z.; KURTHS, J. Complex network approach to characterize the statistical features of the sunspot series. **New Journal of Physics**, v. 16, n. 1, p. 013051, 2014. Citation on page 31.