Biblioteca Digital
F C E N - U B A

BIBLIOTECA CENTRAL LUIS F LELOIR
BIBLIOTECA CENTRAL LELOIR
FACULTAD DE CIENCIAS EXACTAS Y NATURALES UBA

Tesis Doctoral

Métricas de complejidad de grafos para clasificación en múltiples dominios

Tabacman, Maximiliano

2016-05-02

Este documento forma parte de la colección de tesis doctorales y de maestría de la Biblioteca Central Dr. Luis Federico Leloir, disponible en digital.bl.fcen.uba.ar. Su utilización debe ser acompañada por la cita bibliográfica con reconocimiento de la fuente.

This document is part of the doctoral theses collection of the Central Library Dr. Luis Federico Leloir, available in digital.bl.fcen.uba.ar. It should be used accompanied by the corresponding citation acknowledging the source.

Cita tipo APA:
Tabacman, Maximiliano. (2016-05-02). Métricas de complejidad de grafos para clasificación en múltiples dominios. Facultad de Ciencias Exactas y Naturales. Universidad de Buenos Aires.

Cita tipo Chicago:
Tabacman, Maximiliano. "Métricas de complejidad de grafos para clasificación en múltiples dominios". Facultad de Ciencias Exactas y Naturales. Universidad de Buenos Aires. 2016-05-02.

EXACTAS UBA
Facultad de Ciencias Exactas y Naturales

UBA
Universidad de Buenos Aires

Dirección: Biblioteca Central Dr. Luis F. Leloir, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires. Intendente Güiraldes 2160 - C1428EGA - Tel. (++54 +11) 4789-9293

Contacto: digital@bl.fcen.uba.ar

Métricas de complejidad de grafos para clasificación en múltiples dominios

El objetivo de esta tesis es investigar y ofrecer nuevas ideas para clasificar redes complejas basadas en sus propiedades topológicas de gran escala.

Estamos interesados en estudiar las propiedades de redes de diferentes dominios, y encontrar características comunes que son compartidas por las redes en cada una de ellas. Con las mismas pretendemos desarrollar algoritmos y técnicas que aprovechen los elementos comunes, para automatizar la clasificación de redes en sus respectivos dominios. Esta información también será de utilidad para determinar si una red artificial presenta las características esperadas para el dominio al que pretende pertenecer.

Revisamos un gran número de métricas de complejidad encontradas en la literatura. De ellas, elegimos 13 que luego derivan en un total de 43 métricas. Creamos un conjunto de redes de distintas fuentes, con 240 redes distribuidas en 11 dominios. Aplicamos un análisis exhaustivo a las medidas obtenidas para ellas, analizamos su correlación y distribución estadística, e intentamos predecir las posibilidades de clasificación que tendría un algoritmo automático a la hora de discriminarlas correctamente en sus respectivos dominios.

Intentamos determinar si las mediciones obtenidas pueden ser usadas para discriminar los distintos dominios estudiados. Para ello, presentamos 3 tipos de algoritmos de clasificación de la literatura (*K Nearest Neighbors*, *Support Vector Machines* y el sistema de *Evolutionary Rule Learning* **BioHEL**[9]). Las mediciones son utilizadas como valores de entrada para la tarea de clasificación de distinguir el dominio al que pertenece cada red.

Además, para entender el potencial que tienen para clasificar dominios de redes, usamos el generador de grafos **CiGRAM** para crear un conjunto de 160 redes artificiales distribuidas en 8 dominios a modo de ejemplo. Analizamos su distribución estadística y ejecutamos los experimentos propuestos, donde algunos de los algoritmos logran una precisión de hasta 80%. Adicionalmente, observamos que es fácil interpretar los resultados que ofrece el sistema BioHEL, ya que su salida está compuesta de reglas legibles por una persona, que explican cómo clasificar redes en base a los valores de las métricas calculadas.

Finalmente, aplicamos los experimentos propuestos usando las técnicas de clasificación revisadas, con los valores de métricas calculados para las redes reales elegidas. Observamos una clasificación de menor precisión con respecto a lo obtenido para las redes artificiales, y continuamos con escenarios experimentales adicionales para estudiar posibles mejoras, como eliminación de outliers y particionamiento manual de los conjunto de entrenamiento y prueba. Luego de ejecutarlos, a pesar de que en promedio los resultados siguen siendo peores a los obtenidos para las redes artificiales, un subconjunto de los experimentos presenta una precisión de 80% como habíamos obtenido anteriormente.

Nuestras conclusiones permiten confirmar que es posible clasificar redes de distintos dominios considerando únicamente las mediciones obtenidas de un conjunto específico de métricas de complejidad, y también ofrecemos posibles mejoras a ser aplicadas en investigaciones futuras.

Palabras clave: *grafos*, *redes*, *complejidad*, *clasificación*, *métricas*, *automatización*, *algoritmos*, *múltiples dominios*

Graph Complexity Measures for Multi-Domain Network Classification

The objective of this thesis is to research and offer insights into the possibilities of classifying complex networks based on their large scale topological features.

We are interested in studying the properties of networks from different domains, and find common characteristics that may be shared by the networks in each of them. With them, we aim to develop algorithms and techniques that take advantage of the shared features, to automate the classification of networks to their respective domains. This information would also be useful in determining if an artificial network shares the expected characteristics of its intended domain.

We research and review a large number of complexity measures from the literature. From them, we choose 13 that are then derived to reach a total of 43 measures. We created a network data set from different sources, with 240 networks distributed across 11 domains. Exhaustive analysis is applied to the measurements obtained from them, to analyze their correlation and statistical distribution, and attempt to predict the classification possibilities an automated algorithm might have of correctly discriminating between the domains.

We attempt to determine if the measures computed can be used as discriminators between the domains under study. To that end, we present 3 different classification algorithms from the literature (*K Nearest Neighbors*, *Support Vector Machines* and the *Evolutionary Rule Learning* system **BioHEL**[9]). The measures obtained are used as the input for the classification task of distinguishing the domain to which each network belongs.

Furthermore, to understand their potential ability to correctly classify network domains, we use the **CiGRAM** graph generator to build a set of 160 artificial networks distributed across 8 sample domains. We analyze their statistical distribution, and run the experiments proposed, where some of the algorithms presented achieve up to 80% accuracy. Moreover, we observe the ease of interpretation in the results offered by the BioHEL system, since its output is composed of human readable rules that explain how to classify the networks based on their computed measures.

Finally, we apply the proposed experiments using the classification techniques reviewed to the computed measures of the set of real networks chosen. We observe a lower classification accuracy with regard to the values obtained for the artificial networks, and continue with additional experimental scenarios to study possible improvements, such as outlier elimination, and manual partitioning of the training and testing sets. After running them, we observe that despite the average results are still below those obtained for artificial networks, a subset of the experiments present the previously attained accuracy nearing 80%.

Our conclusions imply that it is possible to classify networks into different domains just by considering the measurements obtained from a specific set of complexity measures, and we also offer a number of possible improvements in future related research.

Keywords: *graphs*, *networks*, *complexity*, *classification*, *measures*, *automation*, *algorithms*, *multi-domain*

Agradecimientos

Completar este trabajo llevó varios años de dedicación, durante los cuales conté con la ayuda de muchas personas, a las que quiero agradecer.

Gracias Nai, por la compañía, por las ideas, por la paciencia. Gracias mamá, papá, hermana, por el aliento para que logre mis objetivos.

Gracias Mercap, la empresa que me enseñó a hacer mi trabajo con alegría. En especial gracias Rubén por compartir tus conocimientos de estadística, imprescindibles en mi tesis.

Gracias a la Universidad de Buenos Aires, a la facultad de Ciencias Exactas y Naturales, por la oportunidad de completar esta carrera con una calidad que me enorgullece. Gracias al Departamento de Computación por los años enseñándome los secretos de mi profesión. Gracias a los grupos de investigación en la Universidad de Newcastle, por recibirme y ayudarme en todo momento. En especial gracias Claire, Daven, Jaume, Pawel por sus aportes a los distintos capítulos de esta tesis.

Gracias Natalio, gracias Irene, por la oportunidad de hacer esta tesis y por guiarme a lo largo del camino.

Contents

Contents **I**

1 Introduction **1**
1.1 Motivation . . . 1
1.2 Objetive . . . 2
1.3 Overview . . . 3

2 Measuring Graph Complexity **7**
2.1 Description . . . 7
2.2 What is a Graph? . . . 7
2.3 Measures and Metrics . . . 11
2.4 Complexity Measures . . . 12
2.4.1 Vertex Measures . . . 13
2.4.2 Graph Measures . . . 19
2.4.3 Entropy Measures . . . 40
2.4.4 Summary of Measures Reviewed . . . 49
2.5 Review . . . 50

3 Brief Overview of Classification Techniques **53**
3.1 Description . . . 53
3.2 Problem Definition . . . 54
3.3 Classification methods . . . 56
3.3.1 K Nearest Neighbors . . . 57
3.3.2 Support Vector Machines . . . 59
3.3.3 Evolutionary Rule Learning . . . 61
3.4 Review . . . 66

4 Network Domains and Complexity Measures **69**
4.1 Description . . . 69

4.2 A selection of networks . . . 69
4.2.1 Assorted Biological . . . 70
4.2.2 Code . . . 73
4.2.3 Collaboration . . . 75
4.2.4 Combinatorial DNA Library Synthesis . . . 78
4.2.5 Communication . . . 80
4.2.6 Infrastructure . . . 83
4.2.7 Metabolic . . . 85
4.2.8 Reaction Dependency Graph . . . 87
4.2.9 Social . . . 89
4.2.10 Species Network Graph . . . 92
4.2.11 Tweet . . . 94
4.3 A selection of measures . . . 96
4.3.1 Correlation . . . 97
4.3.2 Real Time Complexity . . . 103
4.3.3 Statistics per Network Class . . . 104
4.4 Review . . . 130

5 Validating Classification Ability 133
5.1 Description . . . 133
5.2 Creating Artificial Networks . . . 133
5.2.1 CiGRAM . . . 133
5.2.2 Generated Networks in Custom Classes . . . 136
5.3 Generated Networks Statistics . . . 136
5.4 Learning to Classify Artificial Networks . . . 143
5.5 Results for the Artificial Networks . . . 152
5.6 Review . . . 154

6 Experiments 157
6.1 Description . . . 157
6.2 Direct classification . . . 158
6.3 Outlier detection . . . 160
6.4 All instances except outliers . . . 161
6.5 Manual partitioning of the training/testing sets . . . 163
6.6 Outliers as a testing set . . . 170
6.7 Review . . . 171

7 Conclusions and Discussions 175
7.1 Description . . . 175
7.2 Contributions . . . 175

7.3 Findings 176
7.4 Future Work 178

A Appendices **185**
A.1 Manual partitioning 185
A.2 Experimental Training and Testing Accuracies Detail 190

Bibliography **219**

List of Figures **235**

List of Tables **241**

Index **249**

Chapter 1

Introduction

1.1 Motivation

The proposal discussed in the following pages is intended as a work in *Toponomics*, understood as the automated classification of complex networks based on their large scale topological features. The main objective is thus to study the properties of such networks, leading to a variety of characterizations, which could enable the development of algorithms and techniques that take advantage of the features that are shared by all sub-components in each class, to produce an automated classification of a network within a given network type.

Since we intend to develop a study that encompasses different kinds of networks, the expected result is a system that could, in principle, be applicable to various such networks, e.g.:

- Social Networks: suggesting grouping of users, finding equivalent communities in different sites
- Financial Instrument Networks: finding similarities in groups of instruments, studying equivalences between markets, detecting suspicious trades
- Software Code Networks: optimization through module grouping, detection of code generalization opportunities
- Ecology Networks: finding equivalent functions in different habitats, detecting critical resources

- Biological Networks: detecting common properties in different cellular components, tracing evolution, studying protein interaction
- Chemical Networks: detecting equivalent compounds, studying chemical interaction

In order to achieve the mentioned objectives, a series of subjects need to be previously studied. From the state of the art in areas such as graph complexity measuring, artificial graph generation and learning classifier systems, a combination of connected ideas will help us propose new research topics in toponomics.

The concept of graph classification studied is focused on determining the original domain of study of different graphs in different domains, and must be differentiated from other classification tasks, such as classification according to graph type, as presented for example in http://www.graphclasses.org/.

1.2 Objetive

Throughout this work, we study the similarities and differences in several network domains, with the final objective of automating their classification. To achieve this, we propose the use of graph complexity measures, as a means of reducing a graph structure to a single number. It is our intention to determine if, given enough complexity measures that present a low correlation between them, a network can be translated to a set of values which can then be used to determine the problem domain it belongs to.

We will present a comprehensive survey of complexity measures, from different domains and sources, aimed at understanding their potential.

We also review and analyze a set of 11 network domains, with a total of 240 networks. They range from biological and chemical sources, to papers studying mailing connections and social networks, including Twitter interaction. The statistical distribution of selected complexity measures is presented and used as an indicator of the feasibility of discriminating between them.

To assist with our experiments and validate our strategies, we build a set of 160 artificial networks belonging to 8 different classes, tailored to present an equivalent scenario to our real world networks.

Classification algorithms are presented as a technique to explore our objectives, with the use of an experimental framework that ensures comparable results. They include both basic schemes such as K Nearest Neighbors, as

well as stochastic programs that employ machine intelligence to produce human-understandable rules for classification.

After running different experimental proposals, the classification of artificial networks is found to be possible with a 80% accuracy. This value is achieved as well for the real word domains, once specific steps are taken to organize the distribution of training and testing samples in the classification algorithms.

1.3 Overview

The rest of this document is structured as follows:

- Chapter 2 introduces the reader to several relevant concepts in graph theory, and contains a survey of several complexity measures, some of which are later employed in this work.
- Chapter 3 defines the classification problem that is our objective, as well as a set of classification algorithms that will be employed in our experiments.
- Chapter 4 lists the network domains reviewed, with a detail of the networks contained for each, and a sample visualization of representative instances that allows a first glimpse at the differences and similarities both inside the domains as well as between them. It also details the selection of complexity measures that are applied in the rest of this thesis, and presents their statistical distribution and correlation when applied to the different domains under study.
- Chapter 5 presents a technique for building artificial networks with known parameters, which are then employed in running the classification algorithms. We include in this chapter an analysis of the statistical distribution of the complexity measures when computed for the artificial networks, followed by experiments that apply the algorithms selected using an experiment building platform.
- Chapter 6 details a collection of experiments using the real world networks reviewed in Chapter 4. They are built with the same platform and equivalent parameters than for the artificial network to ensure comparable results. Different schemes are proposed to create training and testing sets. The accuracy for the different algorithms is displayed and analyzed.

- Chapter 7 closes this work with a summary of the findings presented in each chapter, and the conclusions that can be obtained from them when compared against our objectives. This chapter also lists a number of interesting topics for future research.

Capítulo 1: Introducción

Motivación

Este trabajo se ubica dentro del área *Toponomics*, entendido como la clasificación automática de redes complejas, basándose en sus caracterśticas topológicas de gran escala. El objetivo principal es el estudio de estas redes, y su caracterización para permitir luego el desarrollo de algoritmos y técnicas que aprovechen estas caracterśticas compartidas en cada clase, para producir clasificaciones automáticas de redes de acuerdo a su tipo.

Como nuestra intención es estudiar redes de diferentes tipos, el resultado esperado es un sistema que podría en principio, ser aplicado a distintas redes, a saber:

- Redes sociales: sugiriendo grupos de usuarios, encontrando comunidades equivalentes en distintos sitios
- Redes de instrumentos financieros: encontrando similitudes en grupos de instrumentos, estudiando equivalencias entre mercados, detectando operaciones sospechosas
- Redes de código fuente: optimización a través de agrupamiento de módulos, detección de oportunidades para generalizar código
- Redes ecológicas: encontrar funciones equivalentes en distintos hábitats, detección de recursos críticos
- Redes biológicas: detección de propiedades comunes en diferentes componentes celulares, rastreos evolutivos, estudio de interacción de proteínas
- Redes químicas: detección de compuestos equivalentes, estudio de interacción de químicos

Para lograr los objetivos mencionados, ciertos temas deben ser estudiados previamente. Desde el estado del arte en áreas como la medición de complejidad en grafos, generación de grafos artificiales y sistemas de aprendizaje automático, la combinacioón de distintas ideas nos permitirá proponer nuevas ideas de investigación en Toponomics.

Objetivo

A lo largo de este trabajo, estudiaremos las similitudes y diferencias en distintos dominios de redes, con el objetivo de automatizar su clasificación. Para lograrlo, proponemos el uso de métricas de complejidad de grafos (definidas en la sección 2.4), para lograr reducir la estructura de un grafo a un único número que lo represente. Nuestra intención es determinar la posibilidad de traducir una red a un conjunto de valores que permita detectar el dominio al que pertenece.

Presentaremos un estudio exhaustivo de métricas de complejidad, de diferentes dominios y fuentes, para intentar entender su potencial.

También analizaremos 240 redes, pertenecientes a 11 dominios distintos. Las mismas abarcan desde redes biológicas y químicas, a investigaciones sobre la conexión entre personas a través del envío de correos electrónicos, así como redes sociales como Twitter. La distribución estadística de métricas elegidas especialmente nos permitirá estudiar la factibilidad de clasificar las redes en los distintos dominios a los que pertenecen.

Para ayudar a nuestros experimentos y validar nuestras estrategias, construimos un conjunto de 160 redes artificiales que pertenecen a 8 dominios, armados especialmente para presentar un escenario equivalente al de las redes reales.

Para explorar nuestros objetivos, presentaremos algoritmos de clasificación, y con la ayuda de un ambiente de experimentación podremos garantizar resultados comparables.

Luego de ejecutar los diferentes experimentos, hemos determinado que las redes artificiales pueden ser clasificadas con un 80% de precisión. El mismo nivel de resultado es posible en las redes reales una vez que se toman medidas especiales para organizar los conjuntos de entrenamiento y prueba en los algoritmos de clasificación.

Chapter 2

Measuring Graph Complexity

2.1 Description

This chapter begins the analysis of the existing formulas and techniques that might help in classifying graphs according to their respective domains of origin.

We will first analyze the basic definitions that allow to understand graphs and their properties. Then we will review a number of formulas that allow the representation of the complex structure of a graph as a single number, or collection of numbers (i.e. one per graph vertex, one per graph edge, etc.).

The different formulas reviewed will provide a range of considerations on what differentiates one graph from another, and will be required to understand the classification techniques proposed in the following chapters.

2.2 What is a Graph?

A graph is basically defined as a set of elements (commonly referred to as *vertices* or *nodes*), some of which are connected between them. These connections are known as *edges* or *links*.

Graphs can be *directed* or *undirected*. In the former edges have a direction, i.e. there is an origin vertex and destination vertex, whereas in the latter there is no such distinction.

Although this chapter attempts to provide all the necessary concepts in graph theory required to understand the rest of this work, there are many

books on the subject that delve on theoretical aspects not reviewed here, such as [91, 22].

The terms graph and network might be used interchangeably throughout this work, although graph will be more commonly chosen in this chapter as a general interconnection of elements. Following chapters, since they analyze structures that have a concrete meaning for the graph components (airports and flight routes, proteins and their interactions, people and friendships, etc.), will instead tend to refer to them as networks.

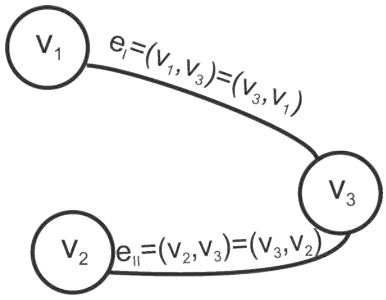


Figure 2.1: Undirected Graph

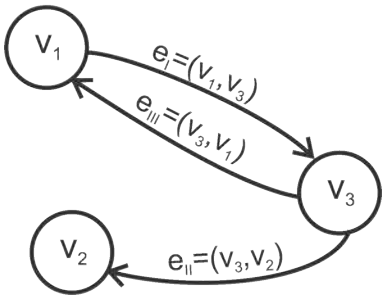


Figure 2.2: Directed Graph

Another classification is that of *weighted* graphs vs. *unweighted* graphs. The first being the case when each edge has a weight or "connectivity", while the second is restricted to representations where there either is a link between two vertices, or there is not.

Definition 1. ***Graph****: A graph G noted as $G(V, E)$, describes a set of vertices $V = \{v_1, \ldots, v_n\}$, connected according to a set of edges $E = \{e_1, \ldots, e_m\}$ (where $e_i = (v_a, v_b)$), with $v_a, v_b \in V$.*

Definition 2. ***Neighbor****: Any vertex v_i connected to vertex v_j by edge (v_i, v_j) or (v_j, v_i) is said to be a neighbor of v_j.*

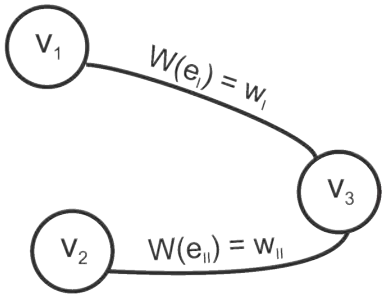


Figure 2.3: Weighted Undirected Graph

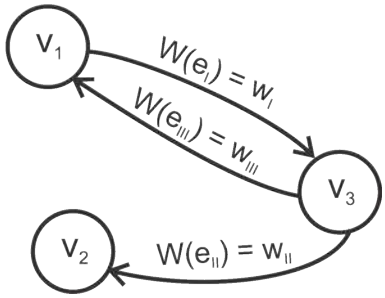


Figure 2.4: Weighted Directed Graph

Definition 3. ***Weighted Graph****: A graph G noted as $G_W(V, E)$, describes a set of vertices V, connected according to a set of edges E, with W a mapping of weight/connectivity for each edge such that $W(e_i) = w_i$.*

Definition 4. ***Degree****: For any given vertex v, the* degree k_v *indicates the number of edges incident to it. Directed graphs also require the distinction between the in-degree k_v^{in} and out-degree k_v^{out}, to specify edges where v is the destination or the origin vertex, respectively.*

Definition 5. ***Path****: A sequence of vertices $v_1, \ldots, v_n$ that can be "traveled" by means of the edges in a graph, in order to reach from v_1 to v_n. Its length is defined as the number of vertices in the path, minus one ($|v_1, \ldots, v_n| - 1$).*

Definition 6. ***Connected Graph****: A given graph G is said to be "connected" when there is a path between any two pair of vertices. If at least one vertex can not be reached from every other one, the graph is "disconnected". Every maximal connected subgraph is referred to as a "connected component".*

Some formulas to be presented later require the graph to be connected, while others include special considerations when it is disconnected. Throughout this work the examples presented in figures 2.5 and 2.6 will be referenced when explaining concepts that are suitable for small graphs.

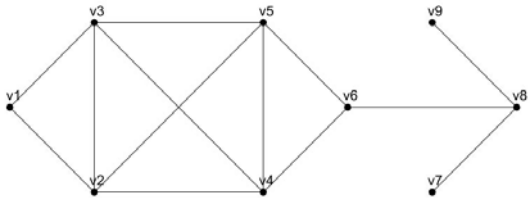


Figure 2.5: Connected Graph Example

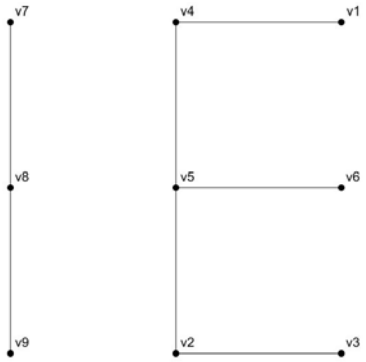


Figure 2.6: Disconnected Graph Example

A common, and useful, practice in graph theory is to represent the edges by means of an *adjacency matrix* A of size $|V| \times |V|$. Each element a_{ij} provides information on the connection between vertices v_i and v_j. Unweighed graphs can only have values 0 and 1, weighed graphs take advantage of this structure to represent the weights. For undirected graphs the adjacency matrix is, of course, symmetric.

	v_1	v_2	v_3
v_1	0	0	1
v_2	0	0	1
v_3	1	1	0

Table 2.1: Undirected Unweighed Graph Adjacency Matrix

	v_1	v_2	v_3
v_1	0	0	1
v_2	0	0	0
v_3	1	1	0

Table 2.2: Directed Unweighed Graph Adjacency Matrix

	v_1	v_2	v_3
v_1	0	0	w_I
v_2	0	0	w_{II}
v_3	w_I	w_{II}	0

Table 2.3: Undirected Weighed Graph Adjacency Matrix

	v_1	v_2	v_3
v_1	0	0	w_I
v_2	0	0	0
v_3	w_{III}	w_{II}	0

Table 2.4: Directed Weighed Graph Adjacency Matrix

2.3 Measures and Metrics

Throughout this work, two distinct type of mathematical functions will be analyzed.

The first is the concept of *measure*. It implies assigning a numerical value to an object under study, in our case *vertices*, *edges*, or *graphs*. This can be as simple as is the case of counting the number of vertices in a graph, and stating that the measure "number of vertices" is computed as $|V|$. Some of them, as will be seen in the rest of this chapter, are inspired by different concepts in algebra, probability, physics, and even biology. Others are complex formulas obtained by combining simpler measures.

A different kind of function that will be of interest are *metrics*. In order for a formula to qualify as such, it must be described as the result of comparing two elements (again, this can mean edges, graphs, etc.) and providing a representation of a distance between them. This implies that a metric $m : A \times A \to \mathbf{R}$ must present properties such as:

1. It must always be positive or zero ($m(a, b) \geq 0 \quad \forall a, b \in A$)
2. It must only be zero when the elements are equivalent ($m(a, b) = 0 \Leftrightarrow a = b$)
3. It must be a symmetric function of its arguments ($m(a, b) = m(b, a)$)

4. The distance between element a and c through b is always at least as that between a and c directly. This is referred to as **triangle inequality** $(m(a,c) \leq m(a,b) + m(b,c))$

2.4 Complexity Measures

Graphs are present in almost every scientific discipline. Sometimes it is plain to see, such as when a graph is used to represent an electric circuit, or the train routes connecting different cities. In other cases, employing graph theory is a decision made by researches to help understand a problem through different angles, as is the case of protein interaction networks or the interactions between a social group.

Several properties of graphs as a structure are of interest first and foremost to graph theory investigators, but specific characteristics are analyzed on occasion by those in other fields, when they may provide insight into their studies. This has led to a wide number of publications where graph measures are presented, with authors ranging from mathematical to life sciences and engineering backgrounds.

One of the main objectives of this thesis is the study of the similarities in graphs, specifically with the idea of finding key characteristics that may allow a classification of the networks represented by field of study. Since the techniques for analysis and processing of graphs are normally restricted to a particular area of interest, it can be argued that a general approach to graph theory will normally leave out some optimization that only applies to a given domain, or risk behaving below the optimum performance[204, 203].

When two different graphs are evaluated using a particular measure, the resulting values can be compared. In a way, the difference between the results of applying a measure to two graphs can be seen as a metric. This metric would indicate how distant the given structures are, with respect to the particular characteristics studied by the chosen measure. Following the ideas in [55, 104, 52], we will be interested in interpreting the value of a measure when applied to a graph as a description of its complexity (understood in a broad sense rather than only in the "computational complexity" sense), hence the term **graph complexity measure**.

Considering the aspects mentioned in the previous paragraphs, the rest of this chapter is aimed at providing a summary of the most relevant vertex and graph measures. For each of them, a summarized description is provided. An idea of computation complexity and underlying structural inspiration is also presented, but only when mentioned in the original paper,

as we will later choose those measures with the simplest and/or better documented ideas and combine them to obtain a more powerful classification tool. For all the items, references are given where the interested reader can obtain further information.

As regards the abbreviations and symbols used to refer to the measures, some of them have been changed from those employed in the papers where they appear. This was done in order to avoid confusion, since many were repeated. Additional concepts required for the definition of particular measures are presented before each case.

First we begin by presenting some vertex measures. They will then be required in the computation of the graph measures that follow.

2.4.1 Vertex Measures

- Degree Centrality (Vertex Degree)[125]: This measure is equal to the degree of the vertex (k_v). According to [125], it is often normalized to help with comparisons, and for directed graphs the measure is split into *In-Degree Centrality* and *Out-Degree Centrality*.

  Definition 7. ***Shortest Path Length(SPL)****: The shortest/geodesic path length (SPL) of v_i to v_j, noted $d(v_i, v_j)$, is the size of the path connecting v_i to v_j that has the fewer number of vertices.*

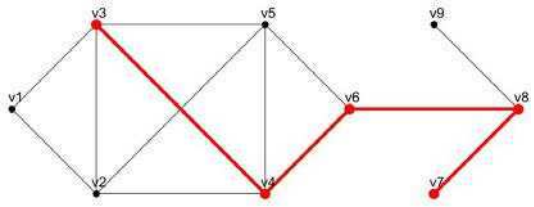


Figure 2.7: Shortest Path Between v_3 and v_7 in an unweighted graph

- Average Vertex Shortest Path Length[85, 51]: This measure, referenced by $\bar{d}_v$, implies obtaining the average of the shortest path length for a given vertex v to all other vertices.

- Betweenness[85, 51, 125, 70, 25, 24]: Betweenness B_v represents the percentage of shortest paths in the network in which v participates.

$$B_v = \sum_{s,t \in V, s \neq t, s \neq v, t \neq v} \frac{\sigma_{st}(v)}{\sigma_{st}} \tag{2.1}$$

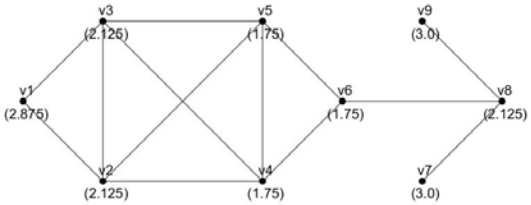


Figure 2.8: Average Shortest Path Lengths

It must be noted that this formula will only consider one (possibly of many) shortest paths in each case. Most implementations will settle for the first shortest path found.

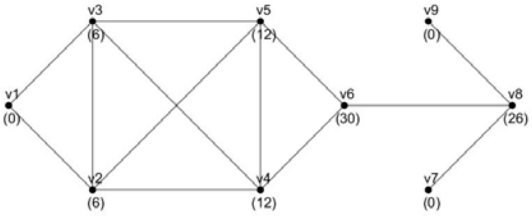


Figure 2.9: Betweenness

Definition 8. ***Maximum Flow / Capacity****: In the specific domain of flow networks, it is important to consider a certain flow (normally expressed as a numeric value) that originates from a specially defined "source" vertex, and moves through the different edges until it reaches a specially defined "sink" vertex. In order to ensure that the flow travels through a "valid" path, a specific restriction is added where each edge is assigned a* ***capacity****. These networks require a flow where incoming values to each node are equal to the outgoing values, and the flow assigned to each edge is below its capacity.*

- Vertex Flow Betweenness Centrality[125, 79]: This measure makes use of the Ford-Fulkerson algorithm for maximum flow, considering the flow that a vertex provides when participating in a path. This requires defining a capacity for each edge in the network (which could be defined as equal to the weights), thus allowing the computation of flows as required by the algorithm. The term m_{jk} represents the maximum flow from v_j to v_k, whereas $m_{jk}(v_i)$ is the maximum flow between them that passes through v_i. The *proportion of network flow*

that depends on v_i is:

$$C'_F(v_i) = \frac{\sum_{j,k(j<k)} m_{jk}(v_i)}{\sum_{j,k(j<k)} m_{jk}} \tag{2.2}$$

It must be noted that this formula assumes the graph is undirected.

- Closeness Centrality (Eccentricity, Status and Centroid Value)[138, 125, 70, 24]: These three measures are closely related, the first being the most commonly used. It indicates the size of the biggest shortest path in the network for a vertex v. Figure 2.10 shows this value for the vertices in the example connected graph. The notation used is:

$$\epsilon(v) = \max_{u \in V}\{d(u,v)\} \tag{2.3}$$

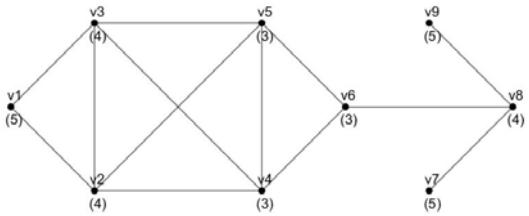


Figure 2.10: Eccentricity

This measure is often used as part of larger, combined measures for networks. This chapter includes a subsection (p. 28) with a selection of such formulas.

- Euclidean Centrality[26]: In case of a network defined as occupying a two dimensional space, with vertices given coordinates, it is possible to analyze the distance of one vertex to another. The *euclidean centrality* of a vertex v is the average distance from v to every other vertex in the network.

  Definition 9. ***Neighborhood****: Γ_v is the set of all neighbors of vertex v. $E(\Gamma_v)$ are the edges between those neighbors. The value $\binom{k_v}{2}$ is the total number of possible edges between them.*

- Local/Vertex Clustering Coefficient[85, 70, 118]: Making use of the previous definition, the clustering coefficient of vertex v, studies the relation between the set of edges and the neighbors of the chosen vertex. It is defined as:

$$\gamma_v = \frac{|E(\Gamma_v)|}{\binom{k_v}{2}} \tag{2.4}$$

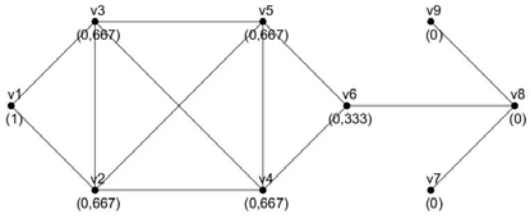


Figure 2.11: Clustering Coefficient in the Connected Graph Example

- Local Cyclic Coefficient[107, 51, 1]: This measure attempts to analyze the cycles related to a given vertex v. The value is defined as:

$$\theta_v = \frac{1}{k_v(k_v-1)} \sum_{j \in V} \sum_{k \in V} \frac{1}{S_{vjk}} a_{vj} a_{vk} \tag{2.5}$$

  where S_{ijk} is the size of the smallest cycle including all three vertices, with the minimum value being 3, and ∞ the maximum, when there is no cycle between them. Since the formula implies considering three different vertices, we will consider that every vertex h with $k_h < 2$ has $\theta_h = 0$.

  The computation requires a non-trivial task: knowing all the cycles in the graph. In [102] a proposal for finding all elementary cycles is presented. It requires time $\mathbf{O}((|V| + |E|) \times (|c_e| + 1))$, where c_e represents the circuits. As a reference, this work mentions that for a directed graph, $|c_e|$ can grow faster than $2^{|V|}$, implying the hard complexity of the overall measure. Mateti and Deo summarize in [129] different efforts in the enumeration of circuits, finally listing Johnson's backtracking algorithm[102] as the fastest. For this reason we will consider this measure as "hard" to compute.

  The **cyclic coefficient** is employed in biological [156, 191, 183, 171, 192] and chemical networks [14], as well as in at least one literature network [39].

- Subgraph Centrality[51, 125, 73]: this value relates a vertex with the number of subgraphs in which it participates in a particular way. Smaller subgraphs have a higher impact on the computation. The formula for vertex i is

$$SC_i = \sum_{j=1}^{Eg} v_j(i)^2 e^{\lambda_j} \tag{2.6}$$

Where Eg refers to the number of eigenvalues, λ_j is the jth eigenvalue and $v_j(i)$ is th ith element for the normalized eigenvector.

- Concentric/Hierarchical Measurements[51, 177]: This group of concepts and related measures are based on a number of common concepts presented in [177]. Before detailing the formulas, we will need to define some ideas. The notions are presented here in the same order as in the original sources. Although currently presented as concentric measurements, it must be noted that the first works where these values are introduced use the term hierarchical instead of concentric.

  Definition 10. ***Concentric/Hierarchical level****: For any given vertex v, the first* concentric/hierarchical level *is defined as the set of immediate neighbors of v, the second concentric level as the immediate neighbors of all nodes in the first concentric level (but without including again those in the first level), the third considers neighbors of the second level (that where not previously considered), an so on.*

  Definition 11. ***Concentric Ring****: For any given vertex v and a level d, the* concentric ring of radius d centered at v $R_d(v)$*, is the set of the nodes in the concentric level d of v. In other words, the concentric ring d of v is defined by nodes at distance exactly d from v.*

  - Concentric number of nodes: The size of a ring $R_d(v)$ is known as the *concentric number of nodes* and is expressed as $n_d(v)$.
  - Concentric number of edges: The count of edges connecting vertices inside ring $R_d(v)$ is known as the *concentric number of edges* and is expressed as $e_d(v)$.
  - Concentric node degree: The measure known as *concentric node degree at level d of v*, expressed as $k_d(v)$ is computed by counting the edges from nodes in the ring $R_d(v)$ to those in $R_{d+1}(v)$. In [51, 177] extensions to this measure to consider weighed edges are included.
  - Concentric clustering coefficient: A notion relating the actual number of edges in a ring against that of the total possible number of them. The formula for the *concentric clustering coefficient* $cc_d(v)$ is:

$$cc_d(v) = 2\frac{e_d(v)}{n_d(v)(n_d(v)-1)} \tag{2.7}$$

– Convergence ratio: this value indicates the relation between the number of edges from one ring to the next with respect to the nodes within the next ring only.

$$C_d(v) = \frac{k_d(v)}{n_{d+1}(v)} \tag{2.8}$$

– Intra-ring node degree: the *intra-ring node degree* provides an average for the degree of nodes in a ring, restricted to the edges in the ring only.

$$A_d(v) = \frac{2e_d(v)}{n_d(v)} \tag{2.9}$$

– Inter-ring node degree: a measure that compares the number of edges that go out to the next level $(d+1)$, against the number of nodes in the current ring (d).

$$E_d(v) = \frac{k_d(v)}{n_d(v)} \tag{2.10}$$

– Concentric common degree: an average degree restricted to edges that participate in a given ring.

$$H_d(v) = \frac{k_{d-1}(v) + k_d(v) + 2e_d(v)}{n_d(v)} \tag{2.11}$$

– r-s clustering coefficient: a variant clustering coefficient formula considering two specific rings r and s, not necessarily consecutive.

$$cc_{r-s}(v) = 2\frac{e_{r-s}(v)}{n_{r-s}(v)(n_{r-s}(v)-1)} \tag{2.12}$$

Where $n_{r-s}(v)$ indicates number of vertices in consecutive rings from $R_r(v)$ to $R_s(v)$ and $e_{r-s}(v)$ is the number of edges among the nodes inside the rings considered.

A summary of these and other related ideas can be found at the author's website, at
`http://cyvision.ifsc.usp.br/Cyvision/?page=CONCENTRIC`.

As regards computational complexity, a non-optimized implementation would require $\mathbf{O}(|E|^d)$ to build the concentric ring $R_d(v)$, although linear implementations are possible. Then, depending on the measure to use, an additional factor of $|E|$ or $|V|$ might be required. For example, obtaining n_2 and e_2 implies $\mathbf{O}(|E|^2)$ and $\mathbf{O}(|E|^3)$ respectively,

which means that $cc_2(v)$ can be computed in $\mathbf{O}(|E|^3)$, or less if specific structures are employed. In [67] it is reported that cc_2 gives an experimental time of $|V|^2$.

Some areas where these ideas have shown interesting results are neuronal[49, 148], collaboration[53] and image[36] networks.

2.4.2 Graph Measures

Vertex/Edge Based

The following properties make use only of the graph components, without need for additional formulas.

- Cardinality of the Vertex Set[32]: One of the simplest conceivable measures, it implies computing the value:

$$N = |V| \tag{2.13}$$

  Although it depends on the specific implementation of the network structure, most often it can be computed in constant time $\mathbf{O}(1)$.

- Cardinality of the Edge Set[32]: Another simple formula, given by:

$$M = |E| \tag{2.14}$$

- Edge Density[70]: A formula that relates how tightly connected a graph is, it can be obtained by:

$$\beta(G) = \frac{M}{\binom{N}{2}} \tag{2.15}$$

Degree Based

These are measures that rely heavily on the concept of degree of the vertices as part of their computation.

- Average Node Degree[85, 70]: All vertices are considered, their degrees recorded, and averaged. This means that

$$\langle k \rangle = \frac{1}{N} \sum_{ij} a_{ij} \tag{2.16}$$

  Time complexity for the required algorithm in most common network representations is $\mathbf{O}(|V|.|E|)$, although linear implementations are possible.

- Maximum Degree[51]: The value k_{max} is computed as

$$k_{max} = \max_i \{k_i\} \tag{2.17}$$

- Variance of the Degree[70]: A measure highly related to the average degree, it is computed by means of the standard variance formula, applied to a graph G:

$$\sigma_k^2(G) = \frac{1}{N-1} \sum_{v \in V} (k_v - \langle k \rangle)^2 \tag{2.18}$$

- Degree Distribution[125]: Given a degree k, this measure indicates the percentage of vertices which have degree k. It is expressed as:

$$P(k) = \frac{|V_k|}{N} \tag{2.19}$$

  with V_k the vertices with degree k.

- Average Degree of the Nearest Neighbors of Vertices with a given Degree[51]: Given a degree k, this measure can be expressed mathematically as:

$$k_{nn}(k) = \sum_{k'} k' P(k'|k) \tag{2.20}$$

  The previous formula requires the definition of conditional degree probability, which in this situation means the percentage of vertices with degree k, which have a neighbor with degree k'. In [51], this sub-formula is presented as:

$$P(k'|k) = \frac{\langle k \rangle \, P(k, k')}{k P(k)} \tag{2.21}$$

  which in turns requires mentioning that $P(k, k')$ is the probability of an edge connecting a vertex with degree k to another with degree k'.

- Degree Heterogeneity[119]: A measure of the spread between the more connected and the less connected vertices in the network.

$$H_G = \frac{\sum_i \sum_j |k_i - k_j| P(k_i) P(k_j)}{\langle k \rangle} \tag{2.22}$$

- Degree Correlation[51]: This is a value mentioned in several papers. Authors mention in [51] that this measure is not always easy to reproduce in artificial networks, when trying to reproduce the characteristics of a studied real networks. For that reason, it is important to take

this into consideration when building random models. The complete formula is based on the Pearson correlation coefficient.

$$CORR = \frac{\frac{1}{M}\sum_{j>i} k_i k_j a_{ij} - \left(\frac{1}{M}\sum_{j>i}\frac{1}{2}(k_i + k_j)a_{ij}\right)^2}{\frac{1}{M}\sum_{j>i}\frac{1}{2}(k_i^2 + k_j^2)a_{ij} - \left(\frac{1}{M}\sum_{j>i}\frac{1}{2}(k_i + k_j)a_{ij}\right)^2} \tag{2.23}$$

- Mean of Absolute Distances[70]: Another measure that applies a probability function to a graph:

$$\tau_k(G) = \frac{1}{N}\sum_{v\in V} |k_v - \langle k\rangle| \tag{2.24}$$

- Zagreb Group Indices[70]: These are two related measures, the first defined for vertices, the second for edges.

$$Z_1(G) = \sum_{v\in V} k_v \tag{2.25}$$

and

$$Z_2(G) = \sum_{(v_i,v_j)\in E} k_{v_i} k_{v_j} \tag{2.26}$$

- Randic Connectivity Index[70]: This simple formula is described by:

$$R(G) = \sum_{(v_i,v_j)\in E} \left(k_{v_i} k_{v_j}\right)^{-\frac{1}{2}} \tag{2.27}$$

- Rich-Club Coefficient[213, 51, 42, 134, 131]:

  The computation of the **Rich-Club Coefficient** is dependent upon the following definition, first presented in [213]:

  Definition 12. ***Rich Club****: Given a degree k, the rich-club for k is the set of vertices with a strictly higher value,*

$$\mathbf{R}(k) = \{v \in V | k_v > k\} \tag{2.28}$$

  To obtain the coefficient, the following formula is used

$$\phi(k) = \frac{1}{|\mathbf{R}(k)|(|\mathbf{R}(k)| - 1)} \sum_{i,j\in\mathbf{R}(k)} a_{ij} \tag{2.29}$$

  This formula implies, once the value k is selected, to iterate over all vertices $v \in V$ in order to obtain those in the "club" and then performing a double sum to consider all connections. The total time is then $\mathbf{O}(|V|) + \mathbf{O}(|V|^2) = \mathbf{O}(|V|^2)$.

An interesting study that can be attempted using this measure is the analysis of the distribution of different values for the coefficient as k increases. Also, in [134] a technique for generating random networks with a given Rich-Club Coefficient is presented, which allows for the creation of models and their later testing.

Being of "easy" calculation, the **Rich-Club Coefficient** is employed in works related to internet [213, 212, 210, 206], biological [42, 208, 33, 123], literary [86], financial [18, 74], social [42, 206] and technological [42, 207] networks.

Shortest-Path Based

In order to obtain the value for any of the formulas in this section, some or all of the shortest paths in the graph need to be previously computed.

- Average Shortest Path Length[85, 51]: This measure (represented simply as d) implies first obtaining an average of the shortest path length $\bar{d}_v$ for each vertex to all other vertices, and then computing the average of this result for all vertices in the graph. Time complexity is $\mathbf{O}(|V|^3)$.

  In Figure 2.12 an example is shown for two networks with a similar number of vertices but different average shortest path length, from those reviewed in the following chapters. They have been colored to show a darker shade and bigger size for vertices with higher degree.

- Average Geodesic Distance[51, 70]: An alternative computation to the Average Shortest Path Length, the formula proposed in this case is:

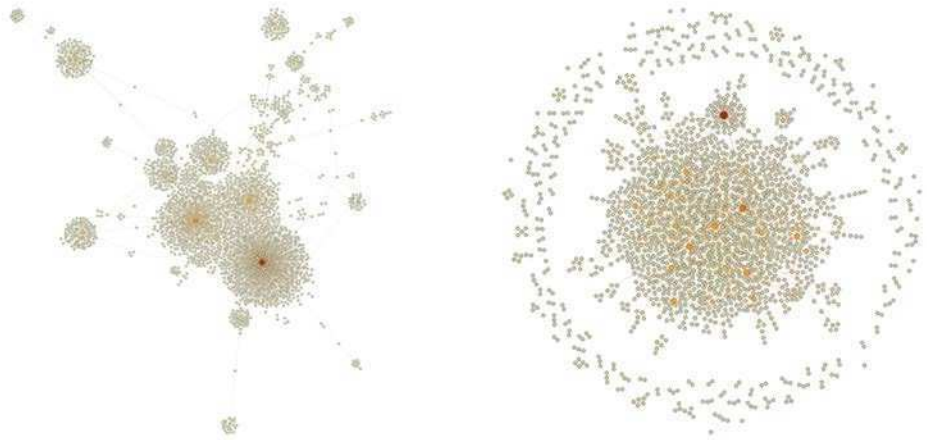
$$\ell = \frac{1}{N(N-1)} \sum_{v_i, v_j \in V, v_i \neq v_j} d(v_i, v_j) \tag{2.30}$$

- Wiener Index[70]: A Classical measure emerging from the chemical sciences, it is defined by the formula:

$$W(G) = \frac{1}{2} \sum_{v_i \in V} \sum_{v_j \in V} d(v_i, v_j) \tag{2.31}$$

  This measure is one of many formulas known as *topological indexes*[194].

- Global Efficiency[51]: A measure proposed as a means to avoid the complications that arise when the Average Geodesic Distance (p. 22) is computed for disconnected graphs. Although this concept is considered for the Average Connected Distance (p. 30) as well, it still



(a) A network with low Average Shortest Path Length, presented in Section 4.2.11

(b) A network with high Average Shortest Path Length, presented in Section 4.2.1

Figure 2.12: Comparison of networks with different values of Average Shortest Path Length. Built using Force Atlas 2 algorithm [100] with Gephi [15]. Nodes represent network vertices, those with higher degree are larger and present a darker shade. Zoom is the closest possible without leaving any node outside. Edges not shown for visualizations where they would block the node distribution.

could lead to oversimplifications when the disconnections are simply ignored. The formula for the **Global Efficiency** attempts to consider all connections, by means of the following equation:

$$F(G) = \frac{1}{N(N-1)} \sum_{v_i, v_j \in V, v_i \neq v_j} \frac{1}{d(v_i, v_j)} \tag{2.32}$$

In order to obtain a comparable measure, the *harmonic mean of the geodesic distances* is used:

$$E_G(G) = \frac{1}{F(G)} \tag{2.33}$$

The computation of this measure implies time complexity $\mathbf{O}(|V|^2)$ due to its iteration over all vertex distances.

This formula is referenced in networks of various types, such as brain [158, 2, 155, 88], social [120] and climate [35] networks.

- Vulnerability[51]: The vulnerability for vertex v, noted as U_v indicates how the loss of said element lowers the global efficiency previously explained. If vertex v is removed, we refer to the new Global Efficiency

of the network G without it and its associated edges as F_v. Then,

$$U_v = \frac{F - F_v}{F} \tag{2.34}$$

The vulnerability of G is finally expressed as

$$U = \max_{v \in V} \{U_v\} \tag{2.35}$$

A simple implementation of the formula, although "easy" to develop, has a time complexity of $\mathbf{O}(|V|^3)$. This could then be improved by storing partial information while iterating the distances for the global efficiency, and consider them at the same time for the adjusted values used in the computation of the vulnerability measure.

The literature currently referencing this measure comprises mainly urban [48, 170], language [7], and traffic [198] networks.

- Complexity Index B[70]: An analysis of the relation between a vertex degree and the distance of its shortest paths. Considering

$$\mu(v_i) = \sum_{v_j \in V} d(v_i, v_j) \tag{2.36}$$

the index is

$$CIB(G) = \sum_{v_i \in V} \frac{k_{v_i}}{\mu(v_i)} \tag{2.37}$$

- Combinatorial Complexity[70]: A measure that combines many of the basic calculations presented here, it sums up the complexity of a graph by analyzing its edges, vertices, and shortest paths. The formula for its computation is:

$$CC(G) = \left(\frac{|V| \times |E|}{|V| + |E|}\right) \sum_{i<j} \sigma_{i,j} \tag{2.38}$$

where $\sigma_{i,j}$ represents the number of shortest paths between v_i and v_j.

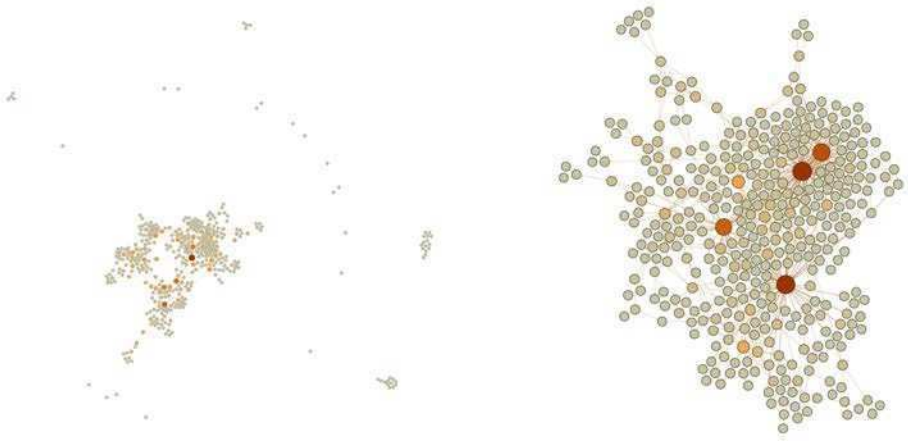
- Characteristic Path Length[118]: Abbreviated as CPL, it is mentioned in [118] as "one of the most important statistics used to measure the shortest distance between each vertex in a graph". First, a set is built containing the Average Vertex Shortest Path Length (p. 13) $\bar{d}_v$ for all vertices $\bar{d}_{v_1}, \ldots, \bar{d}_{v_n}$. CPL is the median of such a set.

Betweenness Based

Some measures, requiring the computation of the betweenness value previously presented, are mentioned next.

- Average Node Betweenness/Betweenness Centrality[85, 51, 125, 70, 24]: The value $B(G)$ equals the average of B_v (p. 13) over all vertices in G.

  In Figure 2.13 an example is shown for two networks from those reviewed in the following chapters, with almost the same number of vertices and edges. One of them shows a low value for Betweenness Centrality, while the other presents a high value. The color shade and size of the vertices has been adjusted to present darker and bigger dots as the value increases (relative to each network).



(a) A network with low Betweenness Centrality, presented in Section 4.2.2

(b) A network with high Betweenness Centrality, presented in Section 4.2.7

Figure 2.13: Comparison of networks with different values of Betweenness Centrality. Node size and darker shade increases according to the observed value for the measure.

- Central Point Dominance[51]: A measure based on that of *betweenness centrality* (p. 25), it is expressed as

$$CPD = \frac{1}{N-1} \sum_{v} (B_{max} - B_v) \tag{2.39}$$

  B_{max} refers to $\max_{v \in V}\{B_v\}$.

- Flow Betweenness Centrality[125, 79]: In [79], it is stated that "there is no reason to believe that people restrict their communication to

the shortest paths in their networks". This leads the mentioned work to look for an alternative to betweenness as a measure of centrality, trying to improve on formulas such as *Central Point Dominance*(p. 25) (referred to in [79] as C_B). As a result, the formula proposed implies: (1) computing the centrality C'_F for all vertices, considering the flow that each provides when participating in a path. (2) With $C'_F(v*)$ defined as the maximum proportion among all possible values in the graph, obtain the centrality measure with:

$$C_F = \frac{\sum_{v_i \in V} [C'_F(v*) - C'_F(v_i)]}{N - 1} \tag{2.40}$$

- Attentive Betweenness Centrality[3]: This measure attempts to consider the benefits and downfalls of *average betweenness centrality*(p. 25), *flow betweenness centrality*(p. 25) and Random Walk Centrality. Already flow betweenness centrality improves upon the average betweenness by considering every edge whether it is included in a shortest path or not, but the maximum flow is always supposed to be available. The implication of this is that a direct path and one with many ramifications along the way will have the same chance of being considered if their flow capacity is the same, while a Random Walk Centrality would give a higher value to former. However, in [3] it is mentioned that Random Walk lacks a desirable quality: forward propagation; that is, the measure allows for a path that goes back and forth, since nothing prevents the randomness from choosing an edge already considered. With all the previous ideas in mind, and also attempting to obtain a computation that is optimal in time and allows working with very large graphs, the Attentive Betweenness Centrality was developed. The algorithm implies a sweep of the graph starting from each node (the *forward step*), and then a reverse analysis to collect all the information (*backward step*). An interesting consequence is that the measure is then not only fast to compute, but easy to multi-thread.

  The forward step is an algorithm that performs a BFS starting from each vertex. At time t, the node v being processed requires the knowledge of whether each neighbor has sent v its "information", has not sent the information yet, or has no information at all to send. The first group is referred to as S_{par} (the parents), the second is S_{sib} (the siblings) and the last one is S_{child} (the children). With x_t being the total information received from the parents, v will forward a fraction

of it ($\alpha x_t/|S_{sib}| + |S_{child}|$) to its siblings first (and at the same time receive information from them, referred to as x_{t+1}) and then in a second step forward a fraction of the total the information to each children ($\alpha(x_t + x_{t+1})/|S_{sib}| + |S_{child}|$). The parameter α is used to measure the relevance of path length (with $\alpha = 0$ only shortest paths are considered).

The backward step is again computed for each node. The BFS tree implied by the forward step is traversed through in reverse. Each node in the bottom layer is given a credit c of 0. Then, for each previous layer each vertex v is considered:

$$c_v = \sum_{u \in U} (1 + c_u) \frac{x_{vu}}{x_u} \tag{2.41}$$

where U is the set of all vertices that interacted with v, x_u is the total flow received by u and x_{vu} is the amount of flow that v provided to u. For each layer all vertices are processed simultaneously. After all nodes have their BFS tree considered, the total score for each node is averaged.

As can be seen from the explanation provided, the time required is $O(|E|)$ for each node, for a total running time of $O(|V|.|E|)$.

Despite its interesting qualities and low complexity, this measure still has, to the writing of this work, not been used in any application outside the original papers. This is not doubt owing to its recent apparition.

- Joint Betweenness Centrality[70, 68]: Given that betweenness centrality (p. 13) considers the shortest paths involving a single node, a possible generalization of this concept was presented in [68]. The new formula expands the single vertex to a set, resulting in:

$$C_{jb}(R) = \sum_{v_i, v_j \in V \ \ R} \frac{\sigma_{v_i,v_j}(R)}{\sigma_{v_i,v_j}} \tag{2.42}$$

As previously mentioned, σ_{v_i,v_j} stands for the number of shortest paths between v_i and v_j, while $\sigma_{v_i,v_j}(R)$ indicates that the paths are restricted to those including all of the elements in R. In the most common case, $|R| = 2$, where the computation is presented as:

$$C_{jb}(v_m, v_n) = \sum_{v_i, v_j \in V, v_i \neq v_j} \frac{\sigma_{v_i,v_j}(v_m, v_n)}{\sigma_{v_i,v_j}} \tag{2.43}$$

An analysis of the complexity for the case where $|R| = 2$ implies considering the time to compute all shortest paths, and then for each pair of nodes, finding the shortest paths that include them. Considering the scenario where a Floyd-Warshall algorithm is used, finding all shortest paths requires $\mathbf{O}(|V|^3)$. Although more optimal configurations could be devised, the basic complexity of the following step implies a four level deep iteration (v_m, v_n, v_i, v_j) and so requires time $\mathbf{O}(|V|^4)$. With this approach, the final complexity would then be $\mathbf{O}(|V|^4)$.

This specific measure has not been used up to now in areas other than those related to biology networks, where it was first defined.

Eccentricity Based

The next two calculations are based on the concept of eccentricity (p.15) defined for vertices.

- Diameter(Maximum Distance between 2 Nodes)[201, 125, 70, 94]: Using the eccentricity vertex measure (p. 15), it is possible to obtain this graph complexity measure, with the formula:

$$\rho(G) = \max_{v \in V} \{\epsilon(v)\} \tag{2.44}$$

  Figure 2.14 allows a visualization of two networks with similar number of vertices and edges. Despite their similar size, one presents a diameter that is more than four times larger than the other. The eccentricity of each vertex determines its size and darker shade in the figure.

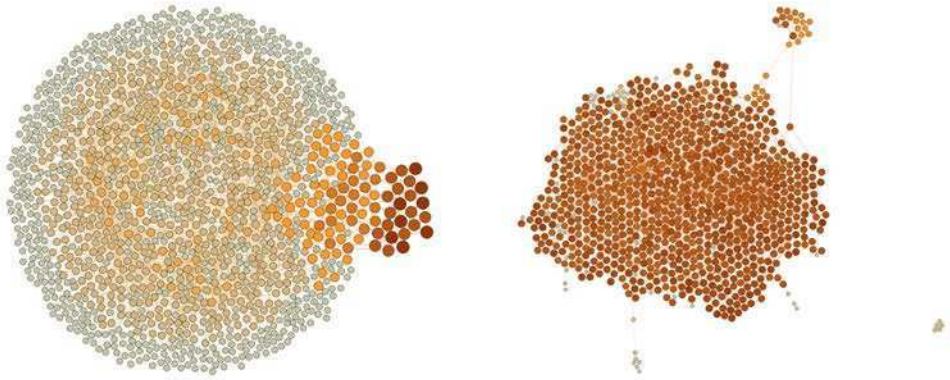
- Radius[70, 94]: In contrast with the computation of the Diameter, the Radius is the minimum eccentricity in the graph:

$$r(G) = \min_{v \in V} \{\epsilon(v)\} \tag{2.45}$$

Other Averaging Measures

Apart from those already presented, some formulas are obtained through the average for all vertices of a given local measure.

- Average Euclidean Centrality[26]: After computing the *euclidean centrality* (p. 15), the average for all nodes can provide a global measure that is useful in comparing networks where the coordinates distribution plays an important part. Iterating over all vertices to compare with ever other one implies a time complexity on the order of $\mathbf{O}(N^2)$.



(a) A network with low Diameter, presented in Section 4.2.4

(b) A network with high Diameter, presented in Section 4.2.7

Figure 2.14: Comparison of networks with different values of Diameter

- (Graph) Clustering Coefficient[118]: This measure has a strong resemblance to the CPL. Its original definition states it as "the extent to which vertices adjacent to any vertex are adjacent to each other". The clustering coefficient γ of a graph G is the average of the vertex measure γ_v previously presented (p. 15), for all vertices $v \in V$.

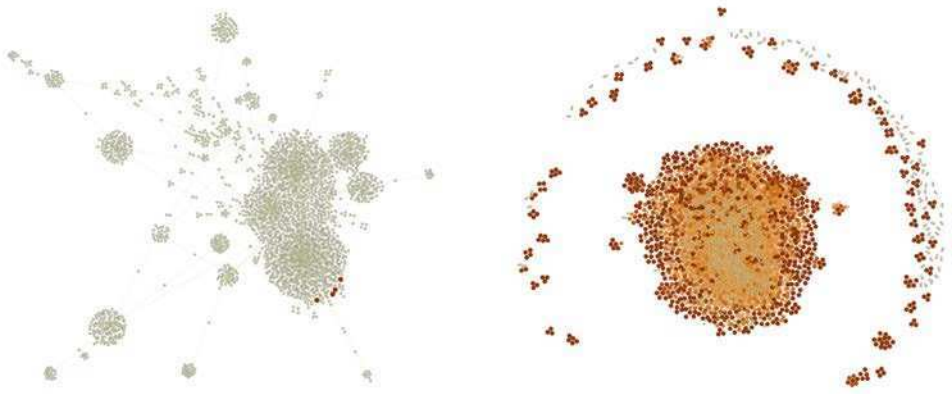
  As an example of the effect of this measure on the structure of a network, we present in Figure 2.15 the visualization for two networks reviewed in the following chapters. While they share the same number of vertices, one of them has a larger Average Clustering Coefficient. The vertex clustering coefficient is used in the visualization to determine the relative size and darker shade of each vertex.

- Edge Clustering Coefficient[51, 177]: Similar to betweenness centrality, this measure is computed for each edge (i, j) in the graph. Interpreting Z_{ij} as the number of "triangles" present in the graph which include (i, j), the idea is to obtain the value C_{ec} by averaging, over all vertices $v \in V$:

$$C_{ec,ij} = \frac{Z_{ij} + 1}{min(k_i - 1, k_j - 1)} \tag{2.46}$$

  In [51], time complexity is mentioned as $\mathbf{O}(\frac{|E|^4}{|V|^2})$.

- Cyclic Coefficient[107, 51]: This measure attempts to analyze the cycles related to a given vertex v, in order to understand "how cyclic"



(a) A network with low Average Clustering Coefficient, presented in Section 4.2.11

(b) A network with high Average Clustering Coefficient, presented in Section 4.2.9

Figure 2.15: Comparison of networks with different values of Average Clustering Coefficient

the network is. The averaged value for the graph is derived from the computation of the Local Cyclic Coefficient θ_v (p. 16) for all vertices:

$$\Theta = \frac{1}{|V|} \sum_{v \in V} \theta_v \tag{2.47}$$

- Concentric Measurements[51, 177]: by considering the different values proposed in the vertex section regarding concentric concepts (p. 17), a number of averaging measures could be obtained, representing the structure of a graph as regards its concentric configuration. Some examples that can be mentioned are level with maximum average concentric node degree, average convergence ratio considering all vertices and all levels.

- Average Connected Distance[28]: Because the existence of disconnected graphs implies the previous Diameter measure (p. 28) results in an infinite value, in [28] a different proposal is presented. With P defined as the set of pairs (u, v) such that there is a path between them, the Average Connected Distance ACD is computed as the expected length of the shortest path considering all pairs in P.

Additional Measures

Although an understanding of graph complexity can be attempted by means of simple calculations, the results are not always varied enough to allow for

a differentiation between two otherwise equivalent networks. Some works have appeared that require more complex formulas than those reviewed so far.

- Offdiagonal Complexity (OdC)[104, 40]: In [104], this measure is described as: "The diversity of the node-node link correlation matrix". It implies the computation of a structure that, basically, counts the number of vertices with degree m that have neighbors with degree n. The size of the matrix is defined by the maximum degree in the network (k_{max}). Each element c_{mn} is defined as:

$$c_{mn} = \sum_{i \in V} \sum_{j \in V} a_{ij} \delta_{m,k_i} \delta_{n,k_j} H(k_i - k_j) \tag{2.48}$$

  $H(x) = 1$ when $x \leq 0$, 0 otherwise. δ refers to the Kronecker symbol (i.e. $\delta_{a,b} = 1$ if a=b, else $\delta_{a,b} = 0$). As previously mentioned, each value c_{mn} represents the number of vertices with degree m directly connected to a vertex of degree n.

  This structure can then be employed to compute a normalized measure by following these steps:

  1. $\tilde{a}_k = \sum_{i=0}^{k_{max}-k} c_{i,k+i}$ (summation over offdiagonals)

  2. $A = \sum_{i=0}^{k_{max}} \tilde{a}_k$ (total summation)

  3. $a_k = \frac{\tilde{a}_k}{A}$ (normalization)

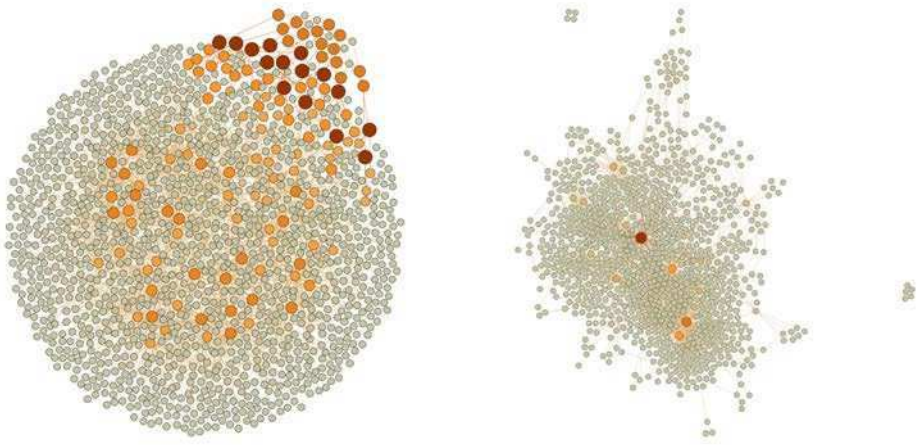
  4. $$OdC = -\left(\sum_{n=0}^{k_{max}} a_k \; log \; a_k \right) \tag{2.49}$$

  This measure has been used (and recommended as giving excellent results) in papers and books that range from communication [174, 139], social [71] and language [38, 46] to biology [99, 136, 137], physics [72, 168] and chemistry [56, 200]. A disadvantage of this measure (as mentioned in [38]) is that the formula does not consider information regarding the strength of the connections (edge weight), limiting the use of the algorithm. Additionally, the procedure as presented in the original paper does not consider directed edges.

  As regards its complexity, it is mentioned in [104] that the computation of this measure is mostly determined by the construction of

the correlation matrix, a step that can be completed in $\mathbf{O}(|E|)$, by iterating each edge only once.

In Figure 2.16 we have selected two networks from those researched for the experiments presented in the following chapters. Both share a similar number of vertices and edges, but differ greatly in the final value obtained when computing their Offdiagonal Complexity.



(a) A network with low Offdiagonal Complexity, presented in Section 4.2.4

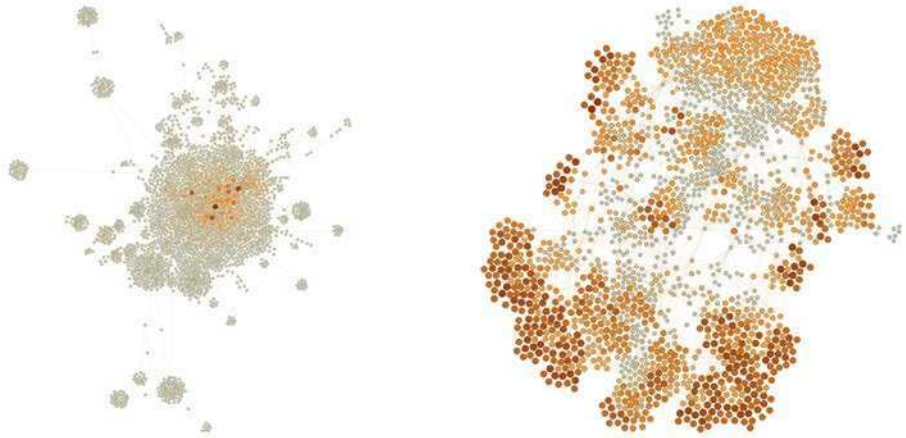
(b) A network with high Offdiagonal Complexity, presented in Section 4.2.7

Figure 2.16: Comparison of networks with different values of Offdiagonal Complexity

- Eigenvector Centrality[85, 125]: This measure requires analyzing the eigenvectors of the adjacency matrix A. The centrality of each vertex is dependent on that of its neighbors, the computation of a normalized eigenvector provides the centrality of each vertex. Since the information to consider requires iterating over all vertices and considering information for all edges, the implementation can be reduced to $\mathbf{O}(|V| + |E|)$.

  We show in Figure 2.17 two networks side by side, from those reviewed in the following chapters. Both share a similar number of vertices and edges, but differ greatly in the average Eigenvector Centrality. The centrality computed for each vertex is used to determine its size and give it a darker shade.

- PageRank[27, 154]: The *page rank* algorithm implies an iterative processing of the edges in a network to find the probability that a random



(a) A network with low Eigenvector Centrality, presented in Section 4.2.11

(b) A network with high Eigenvector Centrality, presented in Section 4.2.4

Figure 2.17: Comparison of networks with different values of Eigenvector Centrality

walk through the graph reaches a given vertex. This means that a value $PR(v)$ is given to each vertex in the network.

To compute it, a time step is initialized at $t = 0$, and $PR(v, 0) = \frac{1}{|V|}$.

Then, each step implies an update:

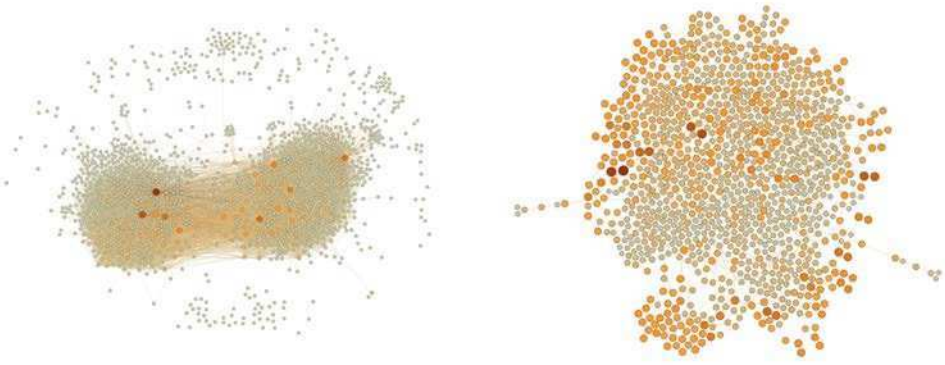
$$PR(v, t+1) = \frac{1 - df}{|V|} + df \sum_{w \in \Gamma_v} \frac{PR(w, t)}{k_w^{out}} \tag{2.50}$$

Where *df* is a damping factor that represents the loss of interest by the random walker in continuing the process. It is typically configured at 0.85.

The iteration proceeds until the global improvement of the values is below a given threshold.

Despite depending on the specifics of each implementation, the algorithm must account in the worst case for each vertex and edge during each iteration, which implies a complexity of $\mathbf{O}(|E| + |V|)$. This holds as long as the number of iterations is bounded.

Figure 2.18 displays two networks, used in the experiments run in the next chapters, which share a similar number of vertices, but where the network with more edges presents a much higher Pagerank value. Vertices with a darker shade and larger size in the figure indicate those that contribute the most to the final value.



(a) A network with low PageRank, presented in Section 4.2.3

(b) A network with high PageRank, presented in Section 4.2.4

Figure 2.18: Comparison of networks with different values of PageRank

- Chromatic Number[138]: This well known concept implies assigning a "color" to each vertex, so that no two adjacent vertices share the same color. If n colors are used in total, the mapping is known as an *n-coloring*. The minimum possible n that still avoids coloring adjacent vertices with the same color is referred to as χ, or *Chromatic Number*. Obtaining this optimum value requires solving an NP-hard problem.

- Chromatic Information Content[138]: Once the minimum number of colors to cover a graph is known via the computation of the chromatic number (p. 34), a further optimization can be achieved by trying to obtain a coloring that also distributes all χ colors amongst the vertices in a way that minimizes the entropy of such distribution. In order to compute such a value, or **chromatic information content**, the following formula is defined:

$$I_c(G) = \min_{\hat{V}} \left\{ -\sum_{i=1}^{\chi} \frac{n_i(\hat{V})}{n} log \frac{n_i(\hat{V})}{n} \right\} \tag{2.51}$$

  Where $\hat{V}$ is some distribution of the vertices in color classes to form a minimum coloring, expressed as $\hat{V} = \{V_i\}_{i=1}^{\chi}$. Each V_i contains all vertices with color i in the distribution being analyzed, and $n_i(\hat{V}) = |V_i|$.

  It is clear from the definition above that computing the chromatic information content optimally requires obtaining **all** possible coloring

with the minimum number of colors, further adding to the complexity of the calculations required. The total computation time requires $\mathbf{O}(2^n n)$. In order to be comparable to other faster methods presented in this work, the computation of this measure should be attempted by approximation, using *sample* coloring distribution of the *estimated* minimum number of required colors. Also, to provide a means of expressing the complexity of a graph by this value, comparisons between graphs should be made not just between their chromatic information contents, but by considering the pair (Chromatic Number, Chromatic Information Content).

The use of this measure in the literature encompasses operation design[211], chemistry[64, 185] and software design[117].

- Number of Spanning Trees[104]: A *spanning tree* T of a graph $G(V, E)$ is a subgraph of G that defines a tree (a connected graph with no cycles) encompassing all vertices $v \in V$, connected with a subset of the original edge set ($E_T \subseteq E$). This measure implies finding all such trees, and returning their number.

- Linear Complexity[104, 143]: This number, in contrast with others analyzed in this work, does not imply a formula tied to a specific algorithm, but is instead a mathematical property defined in terms of the adjacency matrix A of a finite, simple, and undirected graph with n vertices. Considering the product $AX = F$, where X is a vector $x_1, \ldots, x_n$, a set of linear forms F is obtained. The idea is to reduce as much as possible the number of linear forms in the result, without losing information. If $F = f_1, \ldots, f_n$, the linear complexity of the set, $L(F)$, is the length r of the smallest sequence of linear computations that can produce F.

  In [143, 202] many properties are presented, which help compute the linear complexity of a number of graphs, giving also bounds for more general cases. The complexity to obtain this measure can be reduced to $\mathbf{O}(|E| \log(\frac{|V|^2}{|E|}) / \log |V|)$.

  Despite its lack of a defining algorithm, some papers in chemistry [179] and distributed systems [140] have applied the notions of this value to help with bound analysis.

- Cyclomatic Number[104]: Also refered to as the circuit rank, this measure is the result of:

$$CN(G) = |E| - |V| + CC_G \tag{2.52}$$

Here, CC_G is the number of connected components of a graph. For a connected graph, $CC_G = 1$.

- Balaban J Index[70]: This index measure requires the previous calculation of the Cyclomatic Number. Also, we need to define the *Distance or Row Sum* of $v_i \in V$ as DS_i. Finally the index is:

$$J(G) = \frac{M}{CN(G)+1} \sum_{(v_i,v_j)\in E} [DS_i DS_j]^{-\frac{1}{2}} \tag{2.53}$$

- Lempel-Ziv Complexity[32, 31]: Although the *Lempel-Ziv complexity* is originally a complexity measure for strings[113] which attempts to provide a computable value that is related to the Kolmogorov-Chaitin complexity, in [31] a proposal is made to use it for graphs as well. The conversion process is quite straighforward: form a string by concatenating the rows of the adjacency matrix A of a graph G, and apply the already existing algorithms. If using, for example, the standard approach suggested in [106], the value can be obtained in $\mathbf{O}(n)$. This result is in principle valid for unweighted directed graphs, since the "vocabulary" of the string is restricted to 0 and 1, but in the same papers some variants are suggested, and in [31] it is stated that although the computational time may be higher, the addition of more complex elements in the adjacency matrix does not impede the use of the general idea, since they only represent more symbols in the vocabulary.

  Given the simplicity of the idea behind this measure, and its ease of computation, many interesting approaches may be derived, such as analyzing the reduction in complexity after applying transformations to a graph (e.g. reduce a full social graph to its proposed communities). Other interesting properties mentioned in [31] are: *"complementarity (the complexity of a graph is the complexity of its complement) and sample monotonicity (the complexity of a graph is at least as large as the complexity of its most complex induced subgraph)"*.

  Although the original string measure is used in hundreds of derived works, our interest is mainly with the proposal of interpreting the Lempel-Ziv complexity as a measure for graphs. At least as a direct reference, we could not find uses of this idea in specific domains as is the case for other studied measures.

  Apart from the already presented ideas, a lead for future work is mentioned in [31] related to the study of labeled graphs. It is theoretically

possible to differentiate the inherent structural complexity of a graph independently of labels, and that added by the specific labeling. In order to separate this values, it is stated that ideally *all* possible labellings should be analyzed, but the values can be estimated by means of sampling the most relevant labellings, a task well suited for heuristic algorithms.

- Assortativity[51]: When analyzing graphs with "classes" of vertices (e.g. individuals with a given education or economic status, cells from different organs, etc.), an interesting study is that of the comparison of interaction within the class and between different classes. The **assortativity** of a network attempts to explain the level of intra-class interaction, meaning the tendency of a vertex of a given class to have connection with other vertices of the same class.

  In order to define a formula, some preliminary equations must be defined: If C is a matrix of class-class interaction where $c_{s,t}$ represents the number of edges between vertices of class s to those of class t, and $||C||$ is the sum of all its elements, we will refer to $\hat{C}$ as the normalized version of the matrix.

$$\hat{C} = \frac{C}{||C||} \tag{2.54}$$

  The probability of a vertex of class t being connected to one of class s is expressed as $P^{type}(t|s)$. This means that $P^{type}(s|s)$ represents the probability of intra-class connections.

$$P^{type}(t|s) = \frac{\hat{c}_{st}}{\sum_u \hat{c}_{su}} \tag{2.55}$$

  Considering N_T as the total number of classes in the network, $\tilde{Q}$ represents an assortativity coefficient that captures the desired properties.

$$\tilde{Q} = \frac{\sum_s P^{(type)}(s|s) - 1}{N_T - 1} \tag{2.56}$$

  An alternative formulation, which considers giving different weights to the classes depending on the number of vertices for each of them, can be found in [145].

  As regards computation complexity, it can be argued that the matrix C can be built in $\mathbf{O}(|E|)$, and the sums related to $\tilde{Q}$ require at most $\mathbf{O}(|V|^2)$, for a total time of $\mathbf{O}(|E| + |V|^2)$.

  The concept of assortativity is referred to in a large number of studies, mainly in the areas of ecological [121, 176, 108], social [5, 133]

and epidemic [63, 135] networks, but also in collaboration [37] and communication [142] networks.

- Modularity[51, 70, 147]: A critical method when identifying subgraph communities, this measure is presented here as mentioned in [51]. First, c communities must be proposed by grouping the vertices in V. Then we define the matrix M of size $c \times c$. Elements m_{ij} (with $i \neq j$) indicate: for the diagonals the percentage of edges in E that belong inside the community, for the rest the percentage of edges in E that connect both communities. The *Modularity* is then obtained by the following formula:

$$Q = \sum_i [m_{ii} - (\sum_j m_{ij})^2] \tag{2.57}$$

  This means that $Q = TrM - ||M^2||$.

- Bipartivity[51]: A computation of a problem with NP-complete complexity, it is expressed as the minimum value of

$$P = 1 - \frac{\sum_{ij} a_{ij}\delta_{\vartheta(i),\vartheta(j)}}{\sum_{ij} a_{ij}} \tag{2.58}$$

  The fact that there are different values is due to the fact that ϑ is a mapping of vertices to "types", so all possible divisions of the graph are to be considered. δ refers once again to the Kronecker symbol.

- Spectral Density[51, 75, 90]: This formula requires the use of the Dirac delta function $\delta(x)$ [62].

$$\rho(\lambda) = \frac{1}{N}\sum_t \delta(\lambda - \lambda_t) \tag{2.59}$$

  The values λ_t represent the different eigenvalues of the adjacency matrix, and the parameter λ is the eigenvalue which density is to be computed.

- Algebraic Connectivity[51]: This value is obtained by the second eigenvalue of the diagonalized Laplacian Matrix $L = D - A$, where D is a diagonal matrix with $d_{ii} = \sum_j a_{ij}$ and $d_{ij} = 0$ $(\forall i \neq j)$. The technique that gives rise to this measurement is that of *spectral bisection*.

- Edge Reciprocity[51]: A measure specifically aimed at studying directed graphs, it is related to the percentage of edges in E that have

its counterpart also in E, i.e. $(v_i, v_j) \in E \leftrightarrow (v_j, v_i) \in E$. The formula proposed is

$$ER = \frac{\varrho - \overline{a}}{1 - \overline{a}} \tag{2.60}$$

The required values are $\overline{a}$ (the mean value of elements in A) and $\varrho = \frac{\sum_{ij} a_{ij} a_{ji}}{|E|}$. In [51] it is stated that the use of w_{ij} in weighed graphs implies an interesting use of the formula as well.

Definition 13. ***Remaining Degree Distribution****: Instead of counting the number of connections to/from a vertex, the Remaining Degree Distribution excludes one edge. It is computed by:*

$$q(k) = \frac{(k+1)P(k+1)}{\langle k \rangle} \tag{2.61}$$

- Network assortativeness[182]: The normalized correlation function suggested in [182] as a measure of assortativeness for a given graph is

$$NA = \frac{1}{\sigma_q^2} \left[\sum_{j,k=1}^{k_{max}} jkq_c(j,k) - \left(\sum_{j=1}^{k_{max}} jq(j) \right)^2 \right] \tag{2.62}$$

  The resulting values using the mentioned computation presents the properties relevant to the analysis conducted in the mentioned paper.

- Controllability[119]: A measure of the number of vertices in the network that connect to the rest in such a way as to allow to "control" it. In [119] an approximation is presented to compute the value:

$$n_D \approx \exp\left[-\frac{1}{2} \left(1 - \frac{1}{de - 1} \right) \langle k \rangle \right] \tag{2.63}$$

  Where de refers to the degree exponent.

- Critical/Redundant/Ordinary Links Distribution[119]: Three measures that require first the complete analysis proposed in [119]. A critical edge is one which is needed to maintain the *controllability* of the network. Redundant edges have no impact whatsoever in the control outcome. Finally, an ordinary edge does not have a direct impact on the final value, but requires a change in the "driver nodes" that allow for the computed controllability. This means that the percentage of each of these classified edges can provide a structural measure of the graph under study.

2.4.3 Entropy Measures

A considerable number of complexity measures are based on the concept of Entropy. Because of the relevance which this particular approach to complexity has played in recent years, these measures will be presented in a section of their own.

They all borrow from Information Theory the concept of Entropy, as defined by Shannon [173]. For X, a variable (random and discrete) taking values x_i belonging to an alphabet A, and mass probability defined by $p(x_i) = Pr(X = x_i)$, the **entropy** of X is computed with:

$$H(X) = -\sum_{x_i \in A} p(x_i) \log p(x_i) \tag{2.64}$$

Many interesting properties of this formula and derived values can be used in our analysis of graph complexity, and different means of achieving this will be reviewed in the measures listed.

- Codelength[167]: This measure aims at explaining the most convenient way to label the vertices in the network, so as to compute a minimum entropy required to describe them.

  All vertices are assigned a family, and then given a label that is not reused in the same family, but might be reused in another one. Also, each family is given a label, which is not reused for another family. This ensures that each vertex can be described by concatenating its label with that of its family, and ensures the minimum number of labels are used throughout the network.

  Reducing the number of bits required to store the label of a vertex implies in turn that fewer information will be required to describe a path in the network. The *codelength* L represents the lower bound on the average number of bits per step that it takes to describe an infinite random walk on a network partitioned according to a given partition of the vertices.

  To compute it, the following equation is used to obtain the codelength for a given partition M (each family/class of nodes is referred to as a module, and m is the total number of modules):

$$L(M) = q_{out} H(\mathbf{Q}) + \sum_{i=1}^{m} p_w^i H(\mathbf{P}^i) \tag{2.65}$$

To complete the explanation of the previous equation, some additional references are in order:

$$q_{out} = \sum_{i=1}^{m} q_{i,out} \tag{2.66}$$

is the probability that a random walker in the network moves from a vertex belonging to a certain module to one that belongs to another.

The probability $q_{i,out}$ of exiting a particular module i is:

$$q_{i,out} = \tau \frac{n - n_i}{n - 1} \sum_{\alpha \in i} p_\alpha + (1 - \tau) \sum_{\alpha \in i} \sum_{\beta \notin i} p_\alpha w_{\alpha\beta} \tag{2.67}$$

Where, (1) τ is the probability of a random walker to "teleport" to another vertex, and is suggested to be defined as 0.15. (2) α, β are vertices in the module. (3) $w_{\alpha,\beta}$ is the strength of the connection between these vertices. (4) Finally, p_α is the *ergodic visit frequency*, that is, the chance that a random walker chooses to visit vertex α.

The rest of the equations previously presented are completed with the following components:

$$H(\mathbf{Q}) = \sum_{i=1}^{m} \frac{q_{i,out}}{\sum_{j=1}^{m} q_{j,out}} \; log \left(\frac{q_{i,out}}{\sum_{j=1}^{m} q_{j,out}} \right) \tag{2.68}$$

$$p_w^i = q_{i,out} + \sum_{\alpha \in i} p_\alpha \tag{2.69}$$

$$\begin{aligned} H(\mathbf{P}^i) = & \frac{q_{i,out}}{q_{i,out} + \sum_{\beta \in i} p_\beta} \; log \left(\frac{q_{i,out}}{q_{i,out} + \sum_{\beta \in i} p_\beta} \right) \\ & + \sum_{\alpha \in i} \frac{p_\alpha}{q_{i,out} + \sum_{\beta \in i} p_\beta} \; log \left(\frac{p_\alpha}{q_{i,out} + \sum_{\beta \in i} p_\beta} \right) \end{aligned} \tag{2.70}$$

Although a time complexity is not provided, the implementation details provided in the supplementary material of the original paper mention the use of a greedy algorithm followed by a simulating annealing process, which most like requires polynomial time to run.

Further improvements have been made by the original authors, resulting in the possibility of creating a partition of multiple levels. The source code, related papers and additional information can be found at http://www.mapequation.org/code.html.

- Structural Information Content[57, 59, 58]: More a definition for a familiy of formulas than a single equation, the **Structural Information Content** is defined as the entropy of an information functional. The underlying concept mentioned in [57] is that any function that associates a probabilistic distribution to each vertex v in a graph G can then be used to compute the entropy of the network.

$$I_{f^V}(G) = -\sum_{i=1}^{|V|} p^V(v_i) \log\left(p^V(v_i)\right) \tag{2.71}$$

Where the computation of p^V (*vertex probability*) can be easily expressed as:

$$p^V(v_i) = \frac{f^V(v_i)}{\sum_{v_j \in V} f^V(v_j)} \tag{2.72}$$

Then, in [58] a specific function f^V is presented (along with some examples and results).

$$f^V(v_i) = \alpha^{c_1|S_1(v_i,G)|+c_2|S_2(v_i,G)|+\ldots+c_\rho(G)|S_\rho(G)(v_i,G)|} \tag{2.73}$$

In order to understand the previous equation three concepts must be clarified:

1. α is a strictly positive value. It represents an equivalence relation that partitions the vertices in the graph into subsets. This implies that the formula as presented can be then interpreted in a number of ways depending on the clustering provided by this parameter. In [58] it is suggested that α be obtained by considering *class labels* from the graph. It is obviously possible to take advantage of this variable and use the **Structural Information Content** with a metaheuristic to choose the values for α that give similar entropies to graphs that are expected to be equivalent.
2. The set $S_j(v_i, G)$ is referred to as j-sphere of vertex v_i regarding graph G. Its definition is equivalent to the previously presented concept of *concentric ring* (p. 17).
3. The values c_j are another set of tuning parameters for the function. They allow for a prioritization of the structure in the graph. Although the only restriction is that $c_j > 0$, the author suggests in [58] that they are defined in decreasing order as well (e.g. With $\rho(G = 4)$, choose $c_1 = 4, c_2 = 3, c_3 = 2, c_4 = 1$).

The main strength of this measure is clearly the potential for use with different heuristics and machine learning algorithms, made possible by its underlying simplicity. As stated in [58], the total time complexity is $\mathbf{O}(N^3)$.

The known uses for the formula are mainly in biological and chemical networks [70, 60].

An interesting approach to be taken with the idea of this measure is that because of its multiple tuning parameters a multi-step meta-heuristic could be devised. For instance, the optimum values for α could be determined, and then the same could be done with c_j. Given that the computation of the j-spheres/concentric rings is one of the most expensive parts of the formula, and they would not change from one attempt to the next, they can be previously computed in order to reduce as much as possible the total running time of this proposed optimization strategy.

- Entropy of the Degree Distribution[51]: This very simple entropy measure considers the information that the degree of a vertex provides over the overall degree distribution of the network. Needless to say, it overlooks many of the structural properties of the network and in most cases is not to be considered a strong measure of graph complexity.

$$H_{dd} = -\sum_k P(k) \log P(k) \tag{2.74}$$

- Entropy of the Remaining Degree[51, 182]: A subtle change over the previous calculation, this measure considers the remaining degree distribution $q(k)$, already defined. The entropy formula is:

$$H* = -\sum_k q(k) \log q(k) \tag{2.75}$$

- (Automorphism Based) Information Content[138, 59]: This measure requires knowing $Aut(G)$, that is, the automorphism group of the graph under study. It is defined as "the set of all adjacency preserving bijections of V". If $|Aut(G)| = k$, then $\{V_i | 1 \leq i \leq k\}$ is defined as the collection of orbits for said group. If $|V| = n$ and $|V_i| = n_i$ the entropy measure is computed by:

$$I_a(G) = -\sum_{i=1}^{k} \frac{n_i}{n} \log \frac{n_i}{n} \tag{2.76}$$

The formula for $S(v_i, v_j)$ attempts to give an estimate of the amount of information needed to identify one of the shortest paths from v_i to v_j. It is computed with:

$$S(v_i, v_j) = -\log_2 \sum_{\{p(v_i,v_j)\}} \mathbf{P}[p(v_i, v_j)] \tag{2.77}$$

Where $\{p(v_i, v_j)\}$ represents the set of all shortest paths between v_i and v_j. The probability function used, $\mathbf{P}[p(v_i, v_j)]$, expresses the chance of traveling that particular path in a random walk through the graph. It is obtained with:

$$\mathbf{P}[p(v_i, v_j)] = \frac{1}{k_i} \prod_{j \in p(v_i,v_j)} \frac{1}{k_j - 1} \tag{2.78}$$

The set $\{j \in p(v_i, v_j)\}$ is to be interpreted as all vertices in the path, excluding v_i and v_j.

Definition 14. ***Total System Throughflow****: the sum of all directed weights between all vertices in the network;* $TST = \sum_i \sum_j w_{ij}$*. In [201]* T_{ij} *is referred to as the normalized flux because the value is divided by the TST.*

- Entropy of effluxes[201]: This and the next related measures make use of the TST just defined.

$$H(A) = -\sum_i \left(\sum_j T_{ij} \right) \log \sum_j T_{ij} \tag{2.79}$$

  is the entropy of effluxes or uncertainty of sender.

- Entropy of influxes[201]: Analogously to the previous value

$$H(B) = -\sum_j \left(\sum_i T_{ij} \right) \log \sum_i T_{ij} \tag{2.80}$$

  is the entropy of influxes or uncertainty of receiver.

- Joint Entropy[201, 182]: A composition of the both previous computations, analyzing the uncertainty of both sender and receiver;

$$H(A, B) = -\sum_i \sum_j T_{ij} \log T_{ij} \tag{2.81}$$

- Redundancy[201]: This measure arises as a mixture of the conditional entropies in the network. By adding the formula presented in [201] for the uncertainty of sender if the receiver is known, with that of the uncertainty of receiver is the sender is known, the redundancy R(A,B) of the graph is described by:

$$R(A,B) = -\sum_i \sum_j T_{ij} \log \frac{T_{ij}^2}{\sum_k T_{ik} \sum_l T_{lj}} \tag{2.82}$$

- Mutual Information[201]: This combined formula expresses the information about the sender if the receiver is known, but also the information about the receiver if the sender is known.

$$I(A,B) = H(A,B) - R(A,B) \tag{2.83}$$

- Medium Articulation[201]: This measure relies heavily on many of the formulas previously mentioned.

$$MA = I(A,B)R(A,B) \tag{2.84}$$

  In [201], the following equivalent definitions are also presented:

$$MA = I(A,B)(H(A,B) - I(A,B)) \tag{2.85}$$

  and

$$MA = R(A,B)(H(A,B) - R(A,B)) \tag{2.86}$$

- Target and Road Entropies[51]: These two related measures are presented here in one singular entry. They are similar in their approach to the Entropy of effluxes and influxes previously mentioned. The difference is that they do not necessarily specify flow transmission based on edge weight, but rather a nondescript "message flow".

  First, the vertex target and road entropies are defined:

$$T_i = -\sum_{i,j \in V} a_{ji} c_{ij} \log_2 c_{ij} \tag{2.87}$$

$$R_i = -\sum_{i,j \in V} a_{ji} b_{ij} \log_2 b_{ij} \tag{2.88}$$

  Where a_{ij} refers to an element in the adjacency matrix A, c_{ij} represents the fraction of "messages" targeted to i that go through j and b_{ij} is

the fraction that goes through i originating from j. The graph entropy values imply averaging the previous formulas:

$$T = \frac{1}{|V|} = \sum_i T_i \tag{2.89}$$

$$R = \frac{1}{|V|} = \sum_i R_i \tag{2.90}$$

- Information transfer function[182]: A form of mutual information obtained by the conditional probabilities defined in [182], it is summarized as:

$$I(q) = \sum_{k=1}^{k_{max}} \sum_{k'=1}^{k_{max}} q_c(k,k') \log \frac{q_c(k,k')}{q(k)q(k')} \tag{2.91}$$

Definition 15. ***Universal Hierarchical Graph (UHG)****: a reorganization of a given network in a tree-like structure, with the exception that there can be cycles. Apart from the original vertex and edge set, a level set* $\mathbf{L}$ *is required to organize the vertices, and a multi-level function* L *indicates the assignment from vertex to level. Also, the authors in [69] emphasize that a difference between a UHG and a generalized tree is that the former "is allowed to have an arbitrary number of vertices on* $l = 0$*". While there is no proposal for a general transformation from an arbitrary graph to an UHG, the examples presented in the mentioned paper hint that since the levels are expected to be a succession of increasing natural numbers, most domains allow for a simple interpretation of the network information that gives an idea of a proper level set and level function. An interesting property of this graph class is that there is a clear established partition of the nodes.*

- Node entropy of a UHG[69]: Once all components of the UHG are defined, its node entropy is computed using:

$$H^n(G) = -\sum_{i=1}^{|\mathbf{L}|} p_i^n \log_2 p_i^n \tag{2.92}$$

Where p_i^n is the proportion of all vertices that are grouped at level i. Time complexity is presented as $\mathbf{O}(|V|)$.

- Edge entropy of a UHG[69]: Based on the previous definitions, the entropy based on the edges can be defined using:

$$H^e(G) = -\sum_{i=1}^{|\mathbf{L}|} p_i^e \log_2 p_i^e \tag{2.93}$$

In this case, the probability function used requires obtaining $p_i^e = \frac{e_i}{2|E|}$, with e_i representing the number of edges of all vertices on level i. Time complexity is presented as $\mathbf{O}(|V|^2)$.

- Joint entropy of a UHG[69]: A proposed combination of both of the previously presented probability distribution, by considering $p_{ij} = p_i^n p_i^e$ the joint entropy is:

$$H_2(G) = -\sum_{i=1}^{|\mathbf{L}|}\sum_{j=1}^{|\mathbf{L}|} p_{ij}\log_2 p_{ij} \tag{2.94}$$

In order to obtain a discrete joint probability entropy, p_{ij} is to be computed as:

$$p_{ij} = \begin{cases} \frac{p_i^n p_i^e}{\sum_j p_j^n p_j^e} & : i = j, \\ 0 & : i \neq j. \end{cases} \tag{2.95}$$

- Source Entropy of the Edge Set[32]: Based on another formula, that of the first order Shannon entropy, this measure is expressed with:

$$H_{es}(G) = -(\beta\log_2\beta + (1-\beta)\log_2(1-\beta)) \tag{2.96}$$

Where β is the measure for Edge Density(p. 19).

- Information for Adjacency[23]: This measure considers the adjacency matrix A of the graph, and provides a constant value for a given number of vertices N.

$$\begin{aligned} I_{adj} = N^2\log_2 N^2 - 2(N-1)\log_2 2(N-1) \\ - (N^2-2N+2)\log_2(N^2-2N+2) \end{aligned} \tag{2.97}$$

- Information for Incidence[23]: An analogous computation related to the *Information for Adjacency*, the formula for this measure is:

$$\begin{aligned} I_{inc} = N(N-1)\log_2 N(N-1) - 2(N-1)\log_2(N-1) \\ - (N-1)(N-2)\log_2(N-1)(N-2) \end{aligned} \tag{2.98}$$

- Information on Distances[59]: By considering a distance matrix D, where the distances of the shortest paths are presented (d_{ij}), a finite probability scheme related to the graph can be constructed. In [23], a complete analysis on how to obtain the formula is presented.

$$I_D(G) = |V|^2\log(|V|^2) - |V|\log(|V|) - \sum_{i=1}^{\rho(G)} 2k_i\log(2k_i) \tag{2.99}$$

According to [23], this formula provides "a statistical measure for the degree of nonhomogeneity in distribution distances in a given graph". The idea behind its construction is related to the probability of the different nodes to be at their respective distances between each other.

- Information on the Realized Distances[23]: A measure with a final value expressed in [23] as "closer to the Wiener number". The formula uses this value in its expression:

$$\overline{I}_D^W = -\sum_i k_i \frac{i}{W} \log_2 \frac{i}{W} \tag{2.100}$$

In the previous equation, $W = \sum_i ik_i$ is the Wiener number, which the mentioned paper explains as the "total distance of the graph".

Unreviewed Measures

Some measures, which were referred to in the different works researched, are mentioned next along with a short description. They present a novel approach with respect to those previously presented and as such require further study and are a possible source of future work derived from this thesis.

- Average Search Information[51]: Based on the idea that the complexity of a network is related to the ease of navigation through its edges. The formula averages the search information for each combination of vertices $v_i, v_j \in V$.
- Induced Subgraph Complexity[32]: Described in [32] as a measure of "structural diversity", this value requires analyzing the number of different possible subgraphs that could be built using the vertices and edges of the original network under study.
- Treewidth[159]: This measure, as well as Pathwidth and Entanglement mentioned next, are obtained by considering the *branch decomposition*[162, 78, 153] of graphs, which is not reviewed specifically in this thesis but instead presented as a whole topic to be considered in future work in Section 7.4.

  For a given graph G, a game known as Treewidth Game can be analyzed in order to find the minimal number of "cops" k such that the "Cop Player" wins the game $TwG_k(G)$. The Treewidth measure is defined as $k-1$. [159] contains the full definition of the game, including its parameters and rules.

- Pathwidth[159]: A variant of the Treewidth Game mentioned earlier gives rise to this measure, defined as $k-1$, where k is the minimum number of cops required for the Cop Player to win the Pathwidth Game $PwG_k(G)$.
- Entanglement[159]: Another game based measure, $ent(G)$ is explained in [159] as "the minimal number k such that the Cop player has a winning strategy in the game $EG_k(G)$". The interested reader is referred to said work for the full description.
- Locality Index[50]: Based on the matching index presented in [105]. In [50] such index is changed to consider all immediate neighbors of each node, instead of individual edges, giving rise to the *Locality Index*. Its exact computation is not explicitly provided.

2.4.4 Summary of Measures Reviewed

Before finishing the chapter we present a list of all the different network complexity measures that we have mentioned. They are grouped according to the category previously given. The next chapters will make use of a small selection of them to study whether their values can help determine, given a specific network, to which domain it belongs.

Vertex Measures

- Degree Centrality (Vertex Degree)
- Average Vertex Shortest Path Length
- Betweenness
- Vertex Flow Betweenness Centrality
- Closeness Centrality
- Euclidean Centrality
- Local/Vertex Clustering Coefficient
- Local Cyclic Coefficient
- Subgraph Centrality
- Concentric/Hierarchical Measurements
- Concentric number of nodes
- Concentric number of edges
- Concentric node degree
- Concentric clustering coefficient
- Convergence ratio
- Intra-ring node degree
- Inter-ring node degree
- Concentric common degree
- r-s clustering coefficient

Vertex/Edge Based

- Cardinality of the Vertex Set
- Cardinality of the Edge Set
- Edge Density

Degree Based

- Average Node Degree
- Maximum Degree
- Variance of the Degree
- Degree Distribution
- Average Degree of the Nearest Neighbors of Vertices with a given Degree
- Degree Heterogeneity
- Degree Correlation
- Mean of Absolute Distances
- Zagreb Group Indices
- Randic Connectivity Index
- Rich-Club Coefficient

Shortest-Path Based

- Average Shortest Path Length
- Average Geodesic Distance
- Wiener Index
- Global Efficiency
- Vulnerability
- Complexity Index B
- Combinatorial Complexity
- Characteristic Path Length

Betweenness Based

- Average Node Betweenness/Betweenness Centrality
- Central Point Dominance
- Flow Betweenness Centrality
- Attentive Betweenness Centrality
- Joint Betweenness Centrality

Eccentricity Based

- Diameter
- Radius

Other Averaging Measures

- Average Euclidean Centrality
- (Graph) Clustering Coefficient
- Edge Clustering Coefficient
- Cyclic Coefficient
- Concentric Measurements
- Average Connected Distance

Additional Measures

- Offdiagonal Complexity (OdC)
- Eigenvector Centrality
- Chromatic Number
- Chromatic Information Content
- Number of Spanning Trees
- Linear Complexity
- Cyclomatic Number
- Balaban J Index
- Lempel-Ziv Complexity
- Assortativity
- Modularity
- Bipartivity
- Spectral Density
- Algebraic Connectivity
- Edge Reciprocity
- Network assortativeness
- Controllability
- Critical/Redundant/Ordinary Links Distribution

Entropy Measures

- Codelength
- Structural Information Content
- Entropy of the Degree Distribution
- Entropy of the Remaining Degree
- (Automorphism Based) Information Content
- Entropy of effluxes
- Entropy of influxes
- Joint Entropy
- Redundancy
- Mutual Information
- Medium Articulation
- Target and Road Entropies
- Information transfer function
- Node entropy of a UHG
- Edge entropy of a UHG
- Joint entropy of a UHG
- Source Entropy of the Edge Set
- Information for Adjacency
- Information for Incidence
- Information on Distances
- Information on the Realized Distances

Unreviewed Measures

- Average Search Information
- Induced Subgraph Complexity
- Treewidth
- Pathwidth
- Entanglement
- Locality Index

2.5 Review

This chapter provided a meaningful list of complexity formulas that allow us to represent a graph as a single number (or collection of numbers, when using vertex measures).

Section 2.4.4 presented the summary of the formulas reviewed, grouped by considering their focus when analyzing the graph structure.

We have presented complexity measures that are easier to compute, which sometimes implies discarding specific information only present in graphs from a specific domain; and others that make use of additional data not found in all structures, or that implies computing complex iterations over the connections in the graph. All of their benefits and complications should be considered when deciding to apply them to practical tasks.

With these formulas in hand, we can continue in following chapters to

propose experiments where the structure of the graph is reduced to a manageable dimension. This implies that, if we choose a set of measures that are general enough, all graphs from different origins can be expressed in a comparable way. This will be of critical importance in our task of finding the way to classify graphs according to their domain.

Capítulo 2: Midiendo la complejidad de un grafo

Este capítulo comienza el análisis de fórmulas y técnicas existentes que podrían ser de utilidad a la hora de clasificar grafos de acuerdo a su dominio de origen.

Primero analizamos las definiciones básicas que permiten entender a los grafos y sus propiedades. Luego revisamos un gran número de fórmulas que permiten representar la estructura compleja de un grafo como un único número, o colección de números (por ejemplo un número por vértice, o un número por eje). La sección 2.2 revisa los conceptos de grafo dirigido y no dirigido, pesos de un grafo, grado de un nodo, componentes conexas, camino en un grafo y matriz de adyacencia.

La sección 2.4 provee una lista de fórmulas de complejidad que permiten representar un grafo como un único número (o colección de números, en el caso de métricas para vértices). Las distintas fórmulas permiten tener un amplio rango de elementos que podrían permitir diferenciar a los grafos entre si, lo cual será clave para entender las técnicas de clasificación que proponemos en los próximos capítulos.

La sección 2.4.4 presenta el resumen de las fórmulas estudiadas, agrupadas de acuerdo a su criterio para analizar la estructura del grafo. A modo de ejemplo podemos mencionar métricas básicas como la cantidad de vértices y ejes, grado máximo, grado promedio, promedio de camino mínimo, diámetro y radio. También incluimos varias medidas de centralidad y las llamadas de entropía, basadas en los conceptos de teoría de la información.

Hemos presentado métricas de complejidad cuyo cálculo es sencillo, lo cual implica descartar información específica que sólo está disponible en grafos de algunos dominios en particular. Otras requieren información adicional que no estará presente en todas las estructuras, o que implican cálculos complejos a lo largo de un gran número de iteraciones. Todos estos beneficios y complicaciones deberán ser tenidas en cuenta a la hora de aplicarlas con fines prácticos.

Chapter 3

Brief Overview of Classification Techniques

3.1 Description

This chapter provides a detail on the classification problem that we will seek to solve. Basically, we want to know, given a certain graph, to which domain problem it belongs, just by looking at its structure.

The utility of such a system would be two-fold. On one hand, researchers could be working with a network in their field, and use the system to determine if the graph in question is a representative of the domain, to ascertain the best tools to analyze it. On the other hand, developing an automatic domain classification system could help build generic tools to work with networks. Combining both ideas, a subclassifier can be designed: if the proposal of this thesis proves true, a network's structure (described by a series of graph measurements) can determine not only the domain problem to which it belongs, but also the specific type inside its domain. In this case a lot of time could be saved when looking for the best method to tackle each graph problem. An example that this is feasible has been done in [169] for Reaction Dependency Graphs and Species Network Graph, and also in [44] for different network problem instances. Thus in this thesis we seek to extend and generalize these two works for any network.

Just to mention some examples, a graph domain classification system could:

- Distinguish whether some ecological networks behave as a social network (i.e. determine if they share similar features).

- Identify if a road network shares properties of other networks related to it, such as communication towers, airport locations, flight routes, etc.
- Differentiate between protein networks from different species.
- Compare a neuron network from a given subject with others from different development stages to detect health issues.

We will begin the chapter by introducing a number of preliminary concepts, that allow us afterwards to define our classification problem.

Then we proceed to list a selection of classification techniques, presenting for each a detail of their functionality.

3.2 Problem Definition

There is much that can be learned from a network structure such as a graph. A researcher could be interested in finding if a specific property is displayed, check for the predominance of a certain kind of vertex, analyze how the general structure relates to other networks, to name but a few. More can be said if the research in question is related to a specific scientific discipline, as is the case for studies in social networks, protein interaction networks, traffic networks, etc.

The objective of such a learning process can be categorized as either supervised or unsupervised learning, depending on the type of input to provide and the output expected.

Definition 16. ***Supervised learning*** *is the learning process where there is some kind of tutor (automatic or human) that gives the learner direct feedback about the appropriateness of its performance. This is usually achieved by providing the learning system with a training set, experience which has been labeled with the correct response to it, so that the learning system can adjust itself to behave correctly [8].*

In other words, **Supervised Learning** implies the researcher already knows the expected output of a set of inputs, and intends to find a general algorithm that can match those inputs to their known outputs, to be used afterwards on previously unseen cases. The risk of such a process is that this "matching algorithm" could represent only the mapping from the provided samples, which is normally referred to as "overfitting".

Special care must then be used when selecting the samples provided, known as "training set" or "training data". It should encompass a wide

range of inputs that are representative of the general problem that is to be solved. The most common approach is to partition the information available between the "training set" and a "testing set", so that the mapping can be checked against a set of values for which the output is known, but that has not been used during the learning process.

Definition 17. ***Unsupervised learning*** *is characterized by having no performance feedback. In this case, the task of the learning system is to construct some kind of knowledge based only on the flow of experience E, typically trying to identify the regularities existing on E [8].*

Definition 18. ***Classification problem****: Given a set of instances* $I = \{i_1, \ldots, i_n\}$*, each of them labeled with a finite set of classes* $C = \{c_1, \ldots, c_m\}$*, the task of classification is to create a certain theory* T *based on* I *and* C *that, given an unlabeled new instance, can give a prediction of the class of this instance [8].*

A **classification problem** implies the need to separate a group of objects into differentiated classes, based on defining qualities that set them apart, and that make up the essence of each group. Situations ranging from a child's game of recognizing object patterns (squares, circles) to cellular analysis (when discerning between damaged and healthy cells) can be regarded as classification problems.

Many mathematicians and computer scientists try to analyze situations that fit this criteria to model the problem into a clear set of functions and algorithms, to allow for a better chance of finding solutions that are more accurate and easier to obtain.

The mapping implied by a classification problem falls into the category of supervised learning previously defined.

We aim to provide a novel approach to the problem of assigning a class to a given network, from a list of possible classes. As such, it is inherently a classification problem. This can mean either matching graphs to the domain from which they were originated, or categorizing graphs from a given domain between a list of predetermined sub categories. An example of the first proposal is determining whether a graph comes from a social network or a traffic network. For the second proposal we can think for example of determining the species from which a protein interaction network was drafted.

Based on the ideas mentioned in [104], it is our assumption that different networks will present similar values for some complexity measures, while

more varied results for others. So a given graph might look "complex" if analyzing its size ($|V|$ and $|E|$), but have a relatively low Eigenvector centrality. Considering that networks from a given domain sometimes represent different instances of the same problem, it is reasonable to expect that some characteristics will be present in most belonging to a given domain. If we could find these similarities, and they could be observed as values for a measure falling within a given range, then computing these formulas on an unknown network could help us determine the domain problem it belongs to.

The usefulness of such an approach could be furthered. If a researcher working with different techniques to solve a problem, were to know that a group of networks is better handled with one or another of those techniques, and the networks already seen could be analyzed by their complexity measures, a matching between measurements ranges and techniques would allow any future instance to be handled with the best approach from the start.

These ideas can also be applied within a domain, if a sub-classification is required. Biological networks originating from different species might present a certain distribution of results when measured, which could allow a new specimen to be assigned a species based on the values obtained for its corresponding network. Social structures from different environments (relationships at work, sport events, school) could be differentiated, and data from a given network could be compared to see if it "behaves in the expected way of its type".

The previous proposals can be seen as a generalization of the those previously studied for classifying instances of combinatorial optimization problems [44] and for predicting which stochastic simulator to run for instances of synthetic biology networks [169].

3.3 Classification methods

Now that we have settled on considering the task as a classification problem, we can focus on the way in which a system could learn to find similarities in different networks, and propose a classification scheme for future networks whose class we might want to determine.

There are various techniques and algorithmic approaches to classification problems. In this work we will review three of them, and use each in our experiments, to allow their advantages to be combined and to lower the complications any disadvantages may create.

3.3.1 K Nearest Neighbors

One of the simplest classification strategies, **K Nearest Neighbors** requires some preliminary definitions before it can be explained.[17]

Definition 19. ***Nearest Neighbor****: in the context of a classification problem, where each instance is represented by a set of values and a class, a distance metric d can be computed between two instances. This could be as simple as an euclidean distance if all the features are numeric, or could require additional considerations in case of string attributes. Given a set I of instances and a distance function $D(x, y)$, the* nearest neighbor *of an instance i is the instance j where $D(i, j) = min(D(i, x)),\ \forall x \in I$.*

The previous definition can be extended to **k nearest neighbors**, if we consider additional instances, such that if j_1 is the nearest neighbor of i in I, j_2 is the nearest to i when not considering j_1. The same procedure can be followed until we have a set of nearest neighbors $J = j_1, j_2, \ldots, j_k$.

To obtain a classification algorithm, using the ideas mentioned we will apply the procedure described in Algorithm 3.1.

```
k: Number of neighbors
I: Set of known instances
x: Instance with unknown class

FOR (i ∈ I)
        d_i ← D(x, i)
Nearest_Neighbors ← an empty collection
WHILE (Nearest_Neighbors.size < k)
        Nearest_Neighbors ← i ∈ I with minimum d_i
        Remove i from I
Votes ← an empty multiset
FOR (n ∈Nearest_Neighbors)
        Votes.add(n.class)
Suggested_Class ← class in Votes with the highest
   count
RETURN Suggested_Class
```

Algorithm 3.1: K Nearest Neighbor Classification

Given the right distance metric, and a reasonable distribution of the samples, this algorithm easily finds the classes as expected in a classification problem. Figure 3.1 is an example of one such "good case" for the K Nearest Neighbor Classification, where a clear distribution can be identified.

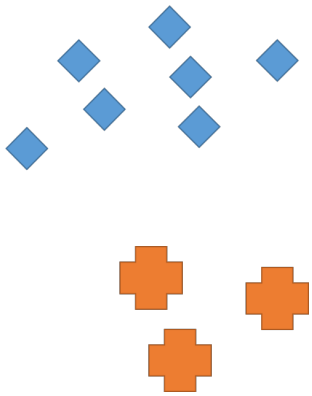


Figure 3.1: Instance distribution where K Nearest Neighbors is a good option. There is a clear distinction between elements of class Diamond and elements of class Cross

As it often happens with algorithms as simple as this one, the relative ease of development is counterbalanced by a number of considerations that have to be addressed. Depending on the distribution of the samples, the number of features to use in the distance metric, and other factors, the class assigned can be far from the real one.

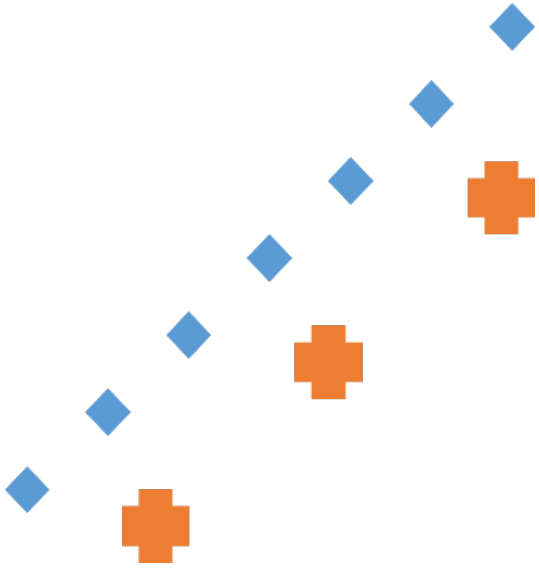


Figure 3.2: Instance distribution where K Nearest Neighbors is not a good option

In figure 3.2 it can be observed that since there are more instances of class *diamond*, and are closely packed with instances of the class *cross*, it is likely that a k nearest neighbors algorithm might assign the *diamond* class to all the instances presented.

There are many proposals for tackling the different shortcomings of this classification technique, from obtaining an "optimum" number of k to use, to suggesting that $k = 1$ should always be preferred. Also some techniques will consider weighted values for the votes offered by the neighbors [47].

Another characteristic that has been studied is the number and relevance of the features in the instances that will allow to make a correct classification. Obtaining a more detailed description of the instances might provide a higher probability of finding the expected result, but because of the need in the technique to assign a distance to every possible neighbor, running times can get prohibitive as the number of features rises. This is why some of the research in this field has targeted the need for better detection of the relevant features [197, 205].

Because of the approach taken by the k nearest neighbor algorithm, there is no generalization that explains the characteristics used in classifying instances. Given a new input, the procedure (at least in its original definition) requires all of the data to be considered each time.

3.3.2 Support Vector Machines

A mathematical approach to classification first proposed in [45] and improved with new ideas in [172], **support vector machines** (SVM for short, but also referred to as support vector networks) attempt to find a division between the samples provided. Furthermore, this division is expressed as an equation that could allow then to assign new instances to a certain region in hyper-dimensional space.

In its simpler form, if instances are expressed as multi-dimensional arrays, and their class is defined as either 1 or -1, a SVM will try and describe each class by its "support vector" which defines the margin that best separates one class from the other. Afterwards, a hyperplane is determined to ensure the maximum distance to the margins. It is the equation for the hyperplane that allows then to classify new inputs as belonging to one or the other class. Figure 3.3 presents the same situation that was mentioned for k nearest neighbor, showing the margins and hyperplane.

The internal working of a SVM is based on the architecture of neural networks, where the neurons have specialized functions to map inputs to the hyper-dimensional space in which the support vectors are expressed.

One of the strengths of SVM is that even when an exact separation of the data into different classes is not possible in the dimensionality considered, the hyperplane resulting from the algorithm will ensure that it is the sep-

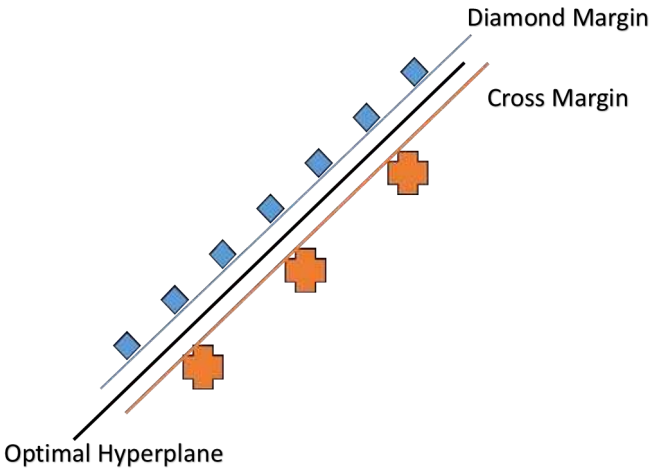


Figure 3.3: SVM components: optimal margin and optimal hyperplane

aration with the minimal number of errors (this is known as a soft margin hyperplane). Figure 3.4 shows an example of this situation.

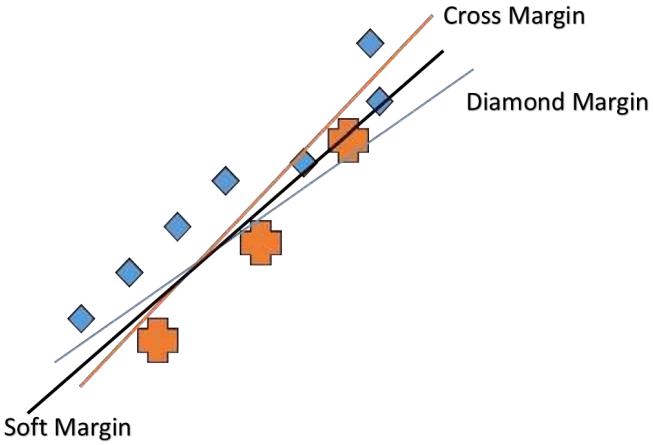


Figure 3.4: Example of a soft margin in SVM

Another good property that arises from the way SVM are designed is that the equation to separate between the classes is linear only in its simplest definition. It is possible to define a more complex function and obtain, for example, a higher degree polynomial to express the margin. Figure 3.5 presents a possible situation where such a scheme could be applied.

To obtain a classification that discriminates between more than 2 classes, a combination of different SVM must be applied, either:

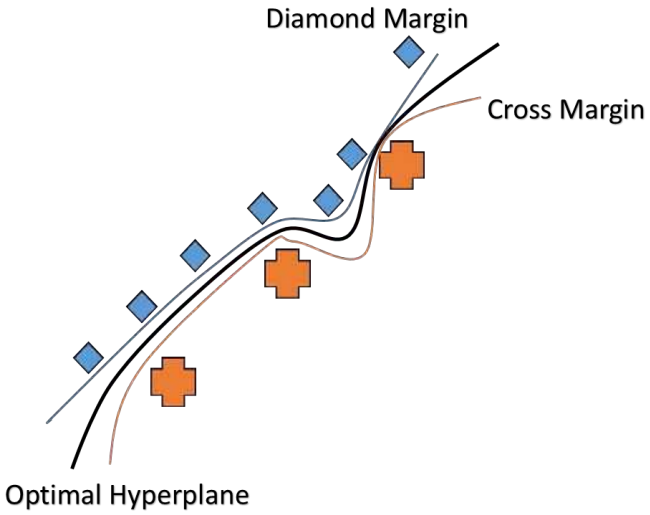


Figure 3.5: Example of a higher degree polynomial used in a SVM

1. Build, for each class c, a SVM that separates between c and $\neg c$.
2. Build one SVM for each pair of classes, then perform a majority voting on the results.

As can be seen from the details mentioned in this section, a support vector machine provides a greater chance of tuning the algorithm to the problem requirements. It provides better reuse than k nearest neighbors because it can be processed once for all the training instances, and then apply only the hyperplane functions to classify new inputs. One disadvantage SVM present is the real possibility of overfitting the solution function if too many samples are considered, or if they are not good representatives of their respective classes.

Additionally, the hyperplane equation is not a good explanation of what is the defining characteristic for separating one class from another, as it represents just a mathematical aggregation of the instances features.

3.3.3 Evolutionary Rule Learning

The last algorithm that will be reviewed implies not only classifying instances, but giving a human-readable explanation on how a future input should be classified.

Learning Classifier Systems were first mentioned around 1978, being [98] the most cited source of its definition. Today they are established as a good option when trying to solve a wide range of problems ([97, 188, 10, 11,

186]). First of all, a LCS is a machine learning technique. This means that the system can, if presented with a task, increase its initial performance at the task as it gains experience. Considerations that arise here are:

1. The task has to be measurable, so that the system's efficiency can be evaluated.

2. Experience has to be defined, and be measurable as well. This can imply registering time, number of attempts at the task, space of inputs already known to the system, etc.

3. The environment associated with performing the task needs to be modeled, so that environmental information can be used as input by the system.

In the particular case of LCS, the task is to classify a set of input data, normally defined as an array of properties (for example classifying animals by means of their height and weight).

Since they were presented, many systems have been designed that fall into this category. The learning procedure is traditionally achieved by considering rules of the form **IF condition THEN action**, where the condition is evaluated against an instance of the input data, and the action implies classifying the current input with the value stated in the rule.

In the case of the animal classification example stated above, we could have rules like *(IF height* $< 1m$ *and weight* $= 3kg$ *THEN cat)* and *(IF height* $< 0.5m$ *and weight* $> 4kg$ *THEN iguana).*

The trick while developing these systems is to find an efficient way of evolving rules like these and reach a set of rules that together classify with high accuracy the whole input space (environment).

What remains to be said about them is that since its beginnings these systems have taken advantage of many techniques from the artificial intelligence area. The specific case of LCS we have chosen follows the *Pittsburgh approach*, which means that a **Genetic Algorithm** (GA) is used as a heuristic to guide the classification process.

Genetic Algorithms are an idea central to **Evolutionary Computation** (EC). The field of EC brings ideas from the biological sciences such as *DNA coding*, *survival of the fittest* and *trait inheritance*. These ideas can be employed by themselves to solve complex problems ([80, 81, 30, 110, 111, 112, 180, 96, 13, 163, 34, 189, 175, 193]). A genetic algorithm is composed of the following elements:

1. An initial population of individuals, with each representing a possible solution to the problem.

2. Generations: the iteration step that identifies one group of individuals from the next (offspring), usually fitter than the parents. In a GA, the "result" is the last generation spawned when the stopping criteria is reached.

3. A criteria for generating offspring from one or more individuals to give rise to the next generation.

4. A criteria for generating small mutations, so as to ensure diversity from one generation of individuals into the next.

5. A fitness criteria, that will ideally be higher in average among individuals from one generation with regards to the previous one.

6. A stopping criteria.

These ideas are applied in the case of LCS to the rules described before. Next is a detail of the definition of GA parameters for an LCS:

1. The population to evolve is a collection of rule sets as those defined before. It is common to represent the rulesets as binary strings, with different sections representing different rules. For example for a rule predicating over 2 properties A and B with a possible value between 0 and 3, 2 bits can be used for each, meaning that 1100 is a ruleset expressing "$A = 3 \& B = 0$".

2. A generation is composed of the current collection of rule sets. It stands that every rule set has at least one rule matching each possible class, so that each individual (rule set) can always classify (even if incorrectly) all of the input data. Some optimizations ignore one of the classes and consider it a Default Class, allowing for smaller rule sets.

3. In the rules used as example above, the offspring criteria could be as simple as taking some rules from two different rule sets and join them to create a new complete rule set, or as complicated as creating new rules with the conditions from one rule set and the actions from another, mixing them with a probability function. Special care must be taken when mixing this rules, since the result must be a valid ruleset.

4. The mutation procedure usually implies a slight change in numerical values, or, if the rules are represented by a byte string, some 0's are changed into 1's or vice versa. As above, the validity of the resulting ruleset must be checked.

5. A fitness formula indicates the ability of the rule set to classify the input data correctly, helping to discard those rule sets that classify incorrectly or are unnecessarily more complex than others already present, allowing the system to evolve more compact and accurate rule sets.

6. Learning Classifier Systems stopping criteria depends on the implementation considered. The model used for this thesis takes a parameter indicating the number of iterations to run, but others could well stop when a certain classification accuracy is reached, or the rule sets presents a certain property.

Aside from the EC component, Learning Classifier Systems require Machine Learning ideas to evaluate and correct the accuracy of the rules. The studied case employs Reinforcement Learning, which means that the input data already contains the expected classification (requiring the data to be supervised when gathered) and the system tries to predict the class based on the rules, to check then against the supplied classification, adjusting its internal values accordingly (depending on whether it was right or not). The specifics of such procedures vary greatly from one LCS to another, and so are out of the scope of this work.

The greatest strength that a LCS will provide to this work with respect to the other two techniques proposed, is the expression of the classification "logic" in the form of human-readable rules.

BioHEL

In particular, we will be working with a specific implementation of a LCS, known as **BioHEL** [9]. It is, more than an algorithm, a complete classification system specifically tailored for large inputs such as the ones researched and mentioned in Chapter 4.

BioHEL is based on GAssist, a Pittsburgh like learning classifier system. GAssist, in turn, is an improvement over GABIL. The most relevant characteristics of these systems are mentioned next, as presented in [190].

GABIL is described in detail in [103]. The features of GABIL that are present in GAssist are:

1. Fitness function (squared accuracy): the fitness of an individual (RuleSet) is computed as $(\frac{\#instances\ correctly\ classified}{total\ \#instances})^2$.

2. Nominal Dataset representation: in the case of classification based on attribute A1 with 4 possible values (X1,X2,X3,X4) and A2 with 2 possible values (Y1,Y2), the rule represented by the binary string 1101 01 implies the predicate "(A1 is X1 or X2 or X4) and (A2 is Y2)".

3. Semantically correct crossover operator: different cut points can be chosen in the parent rules, as long as they are placed in the same position inside the offspring rule.

4. Bit flipping mutation for the nominal representation.

GAssist improves the state of the art for Pittsburgh Learning Classifier Systems by integrating into GABIL the following features:

1. Explicit and static default rule: the best default rule in terms of accuracy and reduction of search space is selected and transparently used throughout the system.

2. Adaptive discretization intervals (ADI): this technique implies an evolving discretization of real-valued attributes, that can vary for each rule and attribute, and which can also split and merge as the system does its job.

3. Incremental learning by alternating strata (ILAS): a windowing technique for generalization and run-time reduction.

4. Bloat control and generalization pressure methods: GAssist presents a rule deletion operator, and a fitness function that considers the content of the individuals in guiding the exploration process.

BioHEL was designed next, with the explicit focus of handling large-scale datasets. We can highlight the following features:

- It applies an Iterative Rule Learning or Separate-and-Conquer approach [82].

- It is based on the Minimum Description Length (MDL) principle [161], a metric which allows to balance the complexity and accuracy of the rules produced.

BioHEL has proved to improve on the results previously produced by GAssist [12, 187].

3.4 Review

This chapter began with a definition of different concepts that are used when working on classification problems.

With them we were able to build a definition of our task: the classification of real world networks according to their domain.

Afterwards, the chapter introduced three techniques that might be employed when solving classification problems: K Nearest Neighbors, Support Vector Machines and Evolutionary Rule Learning.

They will be used in following chapters to attempt to discern that which might differentiate networks in one domain from another.

Capítulo 3: Breve revisión de técnicas de clasificación

Este capítulo describe en detalle del problema de clasificación que pretendemos resolver. Básicamente, queremos saber para un cierto grafo, el dominio al que pertenece, observando únicamente su estructura.

La utilidad de un sistema de estas características radica, por un lado, en la posibilidad de que un investigador pueda determinar si un grafo es "representativo" del dominio al que pertenece, para asegurarse de aplicar las herramientas más eficientes para analizarlo. Por otro lado, desarrollar un sistema de clasificación automática permitiría construir herramientas genéricas para trabajar con redes. De la misma manera, se puede pensar en la confección de un sub-clasificador, las mediciones sobre un grafo podrían permitir no solamente reconocer el dominio al que pertenece, sino el tipo específco dentro del mismo. Este tipo de herramientas ahorrarían tiempo a los investigadores a la hora de buscar la mejor manera de encarar un problema de grafos.

La sección 3.2 presenta las definiciones básicas de un problema de clasificación, aprendizaje supervisado y no supervisado.

En la sección 3.3 detallamos 3 tipos de algoritmos para resolver un problema de clasificación:

- K Nearest Neighbors (KNN)
- Support Vector Machines (SVM)
- Evolutionary Rule Learning (ERL)

Para cada uno revisamos su funcionamiento, y repasamos brevemente ventajas y desventajas de sus enfoques.

En particular, presentamos el sistema de aprendizaje de reglas BioHEL[9]. El mismo resuelve la clasificación a través de un Learning Classifier System (LCS) que incluye un complejo algoritmo genético. Adicionalmente, tiene la ventaja de ofrecer como resultado de su ejecución un conjunto compacto de reglas legibles para una persona, las cuales tienen la forma *(SI* $atributoA < valor1$ *Y* $atributoB > valor2$ *ENTONCES clase X).*

Chapter 4

Network Domains and Complexity Measures

4.1 Description

This chapter has two objectives. The first is to present a list of networks from different domains, and provide a human-made classification that separates them according to their origin. We will intend later to use automatic classification techniques to try to determine if the networks in a domain share some measure of complexity, as presented in Chapter 2.

The second objective will be to choose a subset of the complexity measures in Section 2.4. These values will then be computed for all of the networks selected, and statistical information about them will be presented. By observing the distribution of these values, we will analyze the feasibility of using them as discriminators between the domains under study. Also, in following chapters, these values will be the inputs for a set of classification algorithms, that will attempt to automatically distinguish the domain to which each network belongs.

4.2 A selection of networks

For this project, several networks found in various sources have been compiled and organized according to their domain. This section lists each of the problem domains, along with a summary of the networks that compose them. This summary also indicates the corresponding references that allow the interested reader to find additional information on the network

composition and the context of its construction.

In order to give the reader a better understanding of how these networks "look", each domain will be presented alongside four visualizations of networks in it. These images were obtained by loading the networks using the Gephi framework [15], then coloring the vertices from lighter to darker according to their average degree, and also making vertices with higher average degree slightly larger. Finally, the structure of the networks was built by running the Force Atlas 2 algorithm included with the framework [100]. All visualizations present the closest possible zoom to the structure, without leaving any node outside, which means that more compact networks allow a finer level of detail while networks with vertices that are determined to exist further from a central cluster are presented with lesser detail.

In each subsection, we also present a detail of the networks' number of vertices and edges, sorted by number of vertices.

4.2.1 Assorted Biological

Under this category, we will consider a number of different networks that represent processes in an organism, where vertices commonly represent a gene, protein, transcription factor, etc. and edges reflect an interaction between them ((e.g. a transcription factor activating or inhibiting a gene, a protein-protein interaction, etc).

The categories for *Combinatorial DNA Library Synthesis* and *Metabolic*, presented later, while still belonging to the "Biological" domain, are grouped separately from this one since they offer a higher cohesion in the concepts they represent.

All of the structures used can be obtained at `https://wiki.gephi.org/index.php/Datasets`([83]) and `http://networkrepository.com`([164]), where additional information about their construction can be accessed.

Table 4.1: Assorted Biological Networks

Graph	$\|V\|$	$\|E\|$
NT-bio-MUTAG_g1	23	54
C Elegans	306	2345
NT-bio-celegans	453	2025
NT-bio-celegans-dir	453	2040
NT-bio-diseasome	516	1188

Graph	$\|V\|$	$\|E\|$
Disease	1419	3926
NT-bio-yeast	1458	1948
NT-bio-yeast-protein-inter	1870	4480
Yeast	2361	7182
NT-bio-dmela	7393	25569

- In *NT-bio-celegans* vertices are substrates and the edges are metabolic reactions, with additional edges considered in the alternative network

NT-bio-celegans-dir [65].

- *Disease* and *NT-bio-diseasome* are both related to disorders and disease genes linked by known disordergene associations [89].

- *NT-bio-dmela* presents vertices representing proteins and edges for the protein-protein interactions [178]

- *Yeast* and *NT-bio-yeast-protein-inter* are both Protein-Protein interaction network, where the former was documented in [29] and the latter in [101].

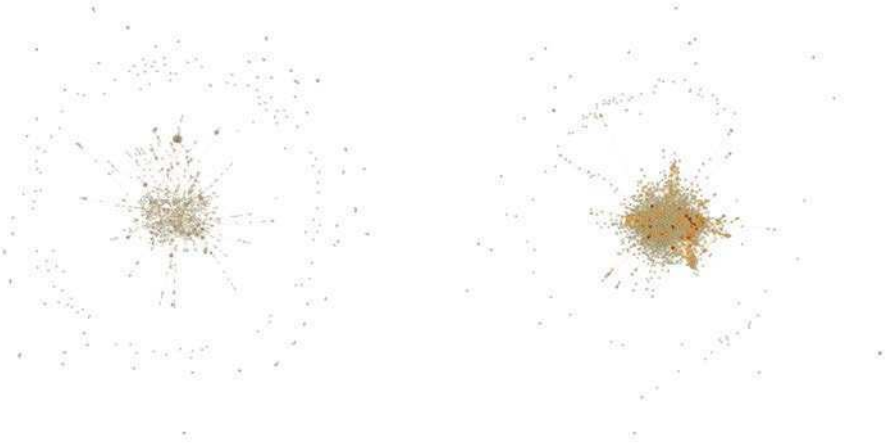
- *C Elegans* is a network from [65] with the information of a neural network in a cell of C Elegans.

- *NT-bio-MUTAG_g1* and *NT-bio-yeast* are included in the biological section of the networks available at [164].

A sample of some of the networks can be seen in Figure 4.1, while the complete list with names, number of edges and vertices can be found in Table 4.1.

(a) Visualization of the "C Elegans" network

(b) Visualization of the "NT-bio-diseasome" network



(c) Visualization of the "NT-bio-yeast-protein-inter" network

(d) Visualization of the "Yeast" network

Figure 4.1: Visualization of four different networks from those selected for the Assorted Biological domain

4.2.2 Code

Code networks are structures related to the evolution of development source code, obtained from different programs and repositories. Vertices are normally users in a collaborative environment, or software packages, while edges indicate a dependency or derivative work.

Table 4.2: Code Networks

Graph	$\|V\|$	$\|E\|$
Word Adjacencies	112	425
Github PHP	301	1071
Github Perl	365	4440
Photoviz	393	621
Github Python	532	2566

Graph	$\|V\|$	$\|E\|$
Github Japan	559	5276
JavaCode	724	1025
Github Europe	2711	11259
Github Profiles	2772	41984
Github Ruby	3742	24571

In particular, the networks included in this domain are:

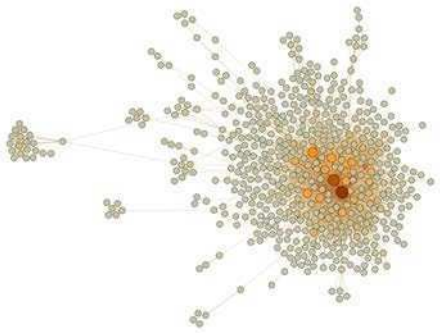
- Github PHP, Github Perl, Github Python, Github Japan, Github Europe, Github Profiles, Github Ruby are project forking relations detailed in `https://lumberjaph.net/github-explorer/`. Vertices are users, and edges represent a project fork from the original user to the one who starts the derived work.

- *Photoviz* is a structure detailing different commits to a SVN source code repository, available at `http://courses.polsys.net/gephi/datasets/`

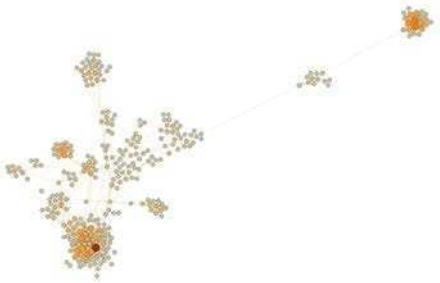
- In *JavaCode*, the structure of a Java program is presented. It can be accessed from [83].

- The *Word Adjacencies* network represents a different kind of "code", since it implies adjacency between common adjectives and nouns [146].

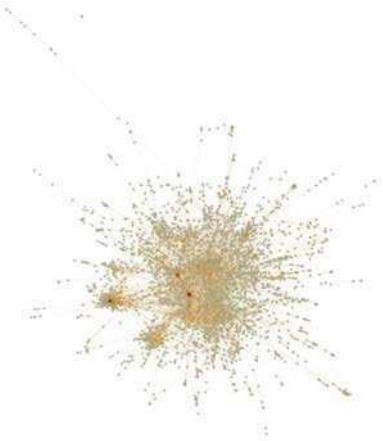
Table 4.2 has the number of vertices and edges for each. Figure 4.2 provides the sample visualizations for this domain.



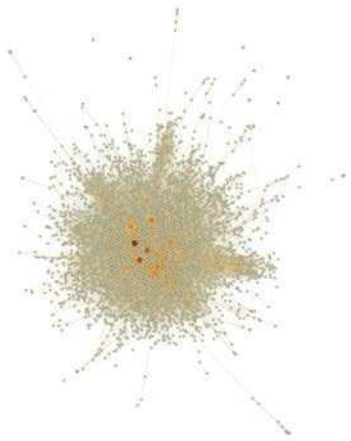
(a) Visualization of the "Github Python" network



(b) Visualization of the "Github PHP" network



(c) Visualization of the "Github Europe" network



(d) Visualization of the "Github Ruby" network

Figure 4.2: Visualization of four different networks from those selected for the Code domain

4.2.3 Collaboration

For the collaboration domain, we have selected networks from different papers and other resources, that represent a collaborative behavior between individuals. This means that vertices will normally represent people, and an edge will imply a joint work or interaction. We have deliberately mixed here structures that represent both scientific collaborations as well as sports and market interactions. It is our intention to observe if these structures share enough common information to be then be classified together in the experiments presented in following chapters.

Table 4.3: Collaboration Networks

Graph	$\|V\|$	$\|E\|$
NT-ca-sandi_auths	86	124
Political Books	105	441
Football	115	613
netscience	379	914
CPAN Authors	840	2248
Political Blogs	1490	19025
NT-ca-CSphd	1882	1740

Graph	$\|V\|$	$\|E\|$
CPAN Distributions	2724	5018
General Relativity	4158	13422
NT-ca-Erdos992	5094	7515
High Energy Physics	12008	237010
Astro Physics	17903	196972
Condensed Matter	21363	91286

- *General Relativity* and *High Energy Physics* indicate scientific collaboration between authors in the General Relativity and Quantum Cosmology, and High Energy Physics - Phenomenology, respecitvely. They were originally presented in [115] (`www.arxiv.org`).

- *Astro Physics* and *Condensed Matter* are built and published in [144], and represent posting preprints on the astrophysics and condensed matter archives at `www.arxiv.org`.

- *CPAN Authors* and *CPAN Distributions* are networks published at `http://cpan-explorer.org/`. The first indicates vertices for developers, linked with when they use the same Perl module. The second indicates Perl modules and their interdependencies.

- *NT-ca-CSphd* is documented in [149, 164], and represents PhD's in computer science.

- *netscience*, from [146], is a co-authoship network of scientists in network theory and experiments.

- *Football* presents a network for the American football games between Division IA colleges during regular season Fall 2000 [84].

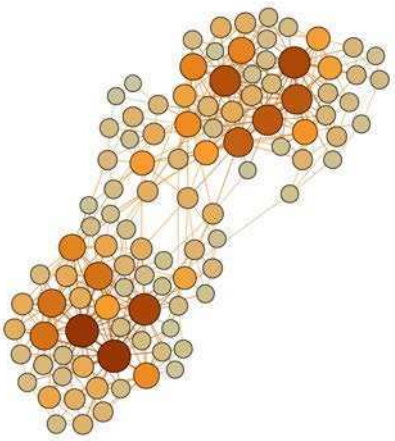
- *Political Blogs* is built using Hyperlinks between political blogs in the United States [4].

- *Political Books* links `Amazon.com` users, creating edges according to the co-purchase of political book during 2004 (`http://www.orgnet.com`).

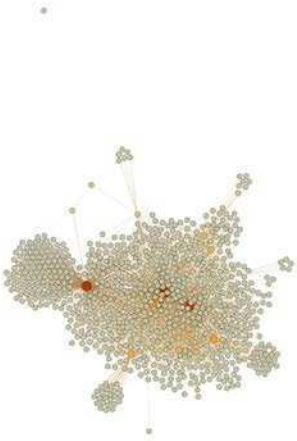
- *NT-ca-Erdos992* is an Erdos collaboration network, presented in [16, 164].

- *NT-ca-sandi_auths* was obtained from the Collaboration section at [164].

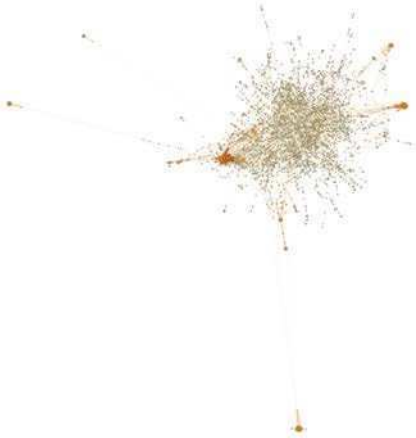
A list with all these structures, sorted by number of vertices, indicating also the number of edges, is presented in Table 4.3. For some examples on the visual representation for some of them, refer to Figure 4.3.



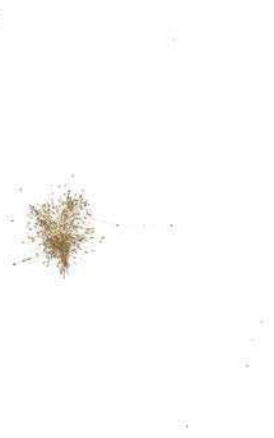
(a) Visualization of the "Political Books" network



(b) Visualization of the "CPAN Authors" network



(c) Visualization of the "General Relativity" network



(d) Visualization of the "NT-ca-Erdos992" network

Figure 4.3: Visualization of four different networks from those selected for the Collaboration domain

4.2.4 Combinatorial DNA Library Synthesis

This domain was obtained from a set of large DNA synthesis (binary) networks employed in the experiments reported in [19] and [20]. Some of them were originally used in [61].

These networks represent steps which could be used then to iteratively build a complex DNA sequence. Vertices represent the different primitive components, or combined components on successive levels, while edges connect components from one step to the combined one in the next.

Figure 4.4 presents a visualization for some of the problems, while Table 4.4 presents the complete list with their respective number of vertices and edges.

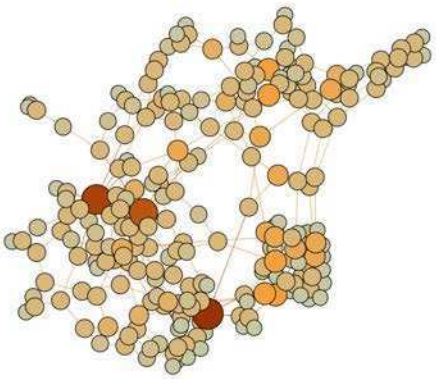
Table 4.4: Combinatorial DNA Library Synthesis Networks

Graph	$\|V\|$	$\|E\|$	Graph	$\|V\|$	$\|E\|$
example-A1-1-2_7	19	21	phagemid-A1-1-4_208	432	624
example-Anew-1-2_7	19	21	uh1-A1-1-3_304	626	912
small_artificial-A1-1-3_11	33	33	uh1-Anew-1-3_344	706	1032
small_artificial-Anew-1-3_14	39	42	uh2-Anew-1-7_344	724	1032
ukb-Anew-1-5_110	241	330	uh2-A1-1-6_515	1066	1545
ukb-A1-1-5_140	301	420	ethz-A1-1-3_690	1393	2070
unott_azurin-A1-1-3_170	359	510	ethz-Anew-1-4_696	1405	2088
unott_azurin-Anew-1-3_170	359	510	iGE2008-Anew-1-5_836	1807	2508
phagemid-Anew-1-4_202	420	606	iGE2008-A1-1-4_839	1813	2517

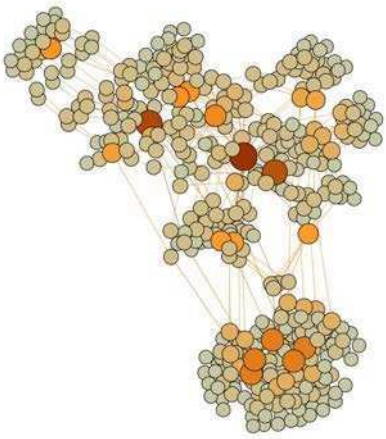
We consider two different sets of networks:

1. small_artificial-A1-1-3_11, example-A1-1-2_7, ukb-A1-1-5_140, unott_azurin-A1-1-3_170, uh1-A1-1-3_304, uh2-A1-1-6_515, ethz-A1-1-3_690 and iGE2008-A1-1-4_839 where built using the assembly algorithm originally published in [61].

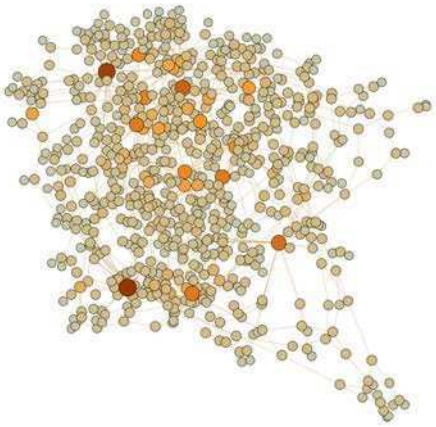
2. small_artificial-Anew-1-3_14, example-Anew-1-2_7, ukb-Anew-1-5_110, unott_azurin-Anew-1-3_170, phagemid-Anew-1-4_202, phagemid-A1-1-4_208, uh1-Anew-1-3_344, uh2-Anew-1-7_344, ethz-Anew-1-4_696 and iGE2008-Anew-1-5_836 are equivalent solutions to the same DNA assembly request built instead with a novel algorithm presented in [20].



(a) Visualization of the "ukb-Anew-1-5_110" network



(b) Visualization of the "phagemid-Anew-1-4_202" network



(c) Visualization of the "uh2-Anew-1-7_344" network

(d) Visualization of the "iGEM_2008-Anew-1-5_836-fa" network

Figure 4.4: Visualization of four different networks from those selected for the Combinatorial DNA Library Synthesis domain

4.2.5 Communication

The communication networks presented here refer specially to situations that might be categorized as social interactions, or even collaborations (in a broad sense), but we will consider only those that are explicit communications in the form of calls, face-to-face or electronic conversations, or mailing. Additionally, some networks presented have been obtained from the communication section of public network repositories (such as `http://networkrepository.com`), to allow later an analysis on the human-made classification of the networks published.

Table 4.5: Communication Networks

Graph	$\|V\|$	$\|E\|$
southernwomen	18	89
infect-hyper	113	2196
enron-only	143	623
radoslaw-email	167	5784
infect-dublin	410	2765
Mails	1133	10902
email-univ	1133	5451

Graph	$\|V\|$	$\|E\|$
fb-messages	1266	6451
reality	6809	7680
PGP Users	10680	24316
wiki-trust-dir	30598	61278
email-EU	32430	54397
Enron Mail	36692	367662
European Mail	265214	420045

In all cases the vertices are people, while the edges represent the communication recorded by each particular network, as detailed next:

- Both *infect-hyper* and *infect-dublin*, presented in [181], are built considering a "proximity contact" between attendees to a conference and an artscience exhibition, respectively.

- *Enron Mail* and *enron-only* are alternative networks built in [116, 41, 164] with information published by the Federal Energy Regulatory Commission during an investigation.

- *email-univ* and *Mails* represent mails between members of a university, where the latter includes edges in both directions [92].

- *European Mail* and *email-EU* are networks published in [115] using mails between members of an European research institution. The second one considers only mutual exchanges to create an edge.

- *radoslaw-email* represents the interaction according to mails between employees of a manufacturing company [132].

- *wiki-trust-dir* considers interactions between Wikipedia users on political pages [124].

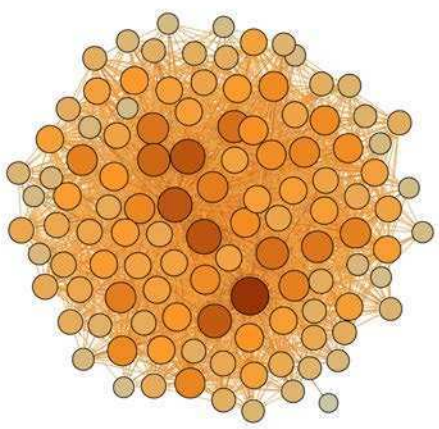
- The *PGP Users* network considers users of the PGP algorithm and their communication. It is presented in [21] and specified to include only the giant connected component of the derived structure.

- *southernwomen* is built according to the participation of a group of women in specific social events [54].

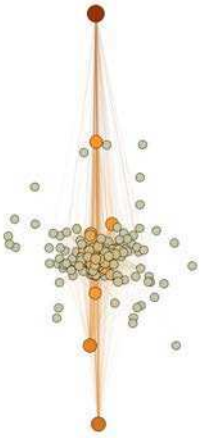
- *reality* was obtained from [66], and represents calls between subjects of a reality experiment.

- *fb-messages* has vertices for different Facebook members, with edges indicating communication between them [152].

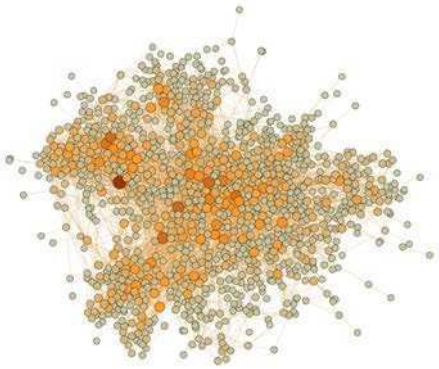
In Table 4.5 all of the networks in this domain are listed, along with the number of vertices and edges for each. Figure 4.5 then presents a sample visualization for four of them.



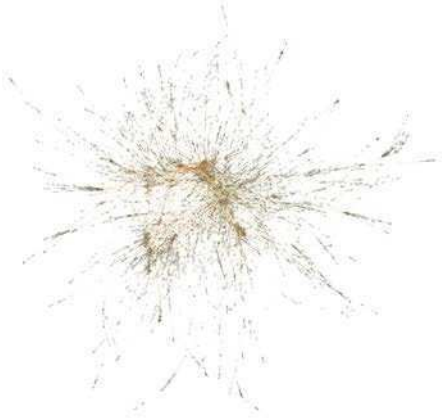
(a) Visualization of the "infect-hyper" network



(b) Visualization of the "radoslaw-email" network



(c) Visualization of the "email-univ" network



(d) Visualization of the "PGP Users" network

Figure 4.5: Visualization of four different networks from those selected for the Communication domain

4.2.6 Infrastructure

The infrastructure domain is a gathering of networks that represents the real world location of different wide spanning projects. They range from Internet routers to Airline routes.

Vertices represent the location of the different objects surveyed by the networks, and the edges represent an existing real connection (cables, roads, planned airplane route, etc.)

Table 4.6: Infrastructure Networks

Graph	$\|V\|$	$\|E\|$
Top-Ireland	7	11
Top-Australia	19	24
Top-Spain	19	31
Top-Belgium	22	25
Top-Canada	23	23
Top-Argelia	30	27
Top-Brazil	31	34
Top-Chile	37	36
Top-France	43	56
Top-Italy	61	75
Top-Armenia	65	59

Graph	$\|V\|$	$\|E\|$
Top-Japan	74	76
Airlines	235	2101
USAir97	332	2126
routers-rf	2113	6632
openflights	2939	30501
Power Grid	4941	6594
WHOIS	7476	56943
Internet Routers	22963	
as-caida2007	26475	53381
internet-as	40164	85123
p2p-gnutella	62561	147878

- A number of networks were obtained from `http://www.topology-zoo.org/index.html`. They represent the result of applying a trace to understand the topology of internet connections in different countries. We have selected, from this survey, the following networks: Top-Ireland, Top-Australia, Top-Spain, Top-Belgium, Top-Canada, Top-Argelia, Top-Brazil, Top-Chile, Top-France, Top-Italy, Top-Armenia and Top-Japan.

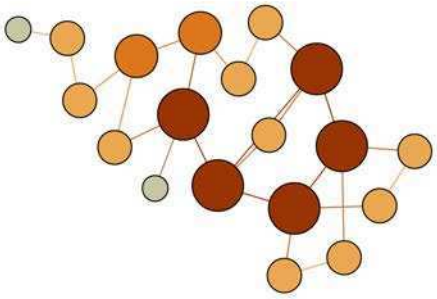
- Some flight routes and airport connections are present in the form of the network *Airlines*, *USAir97*, *openflights*. They were obtained from [83], [164, 43] and [151], respectively.

- Additional internet and intranet structured obtained were *routers-rf* [164, 184], *WHOIS* [164, 122], *Internet Routers* [83], *as-caida2007* and *internet-as*.

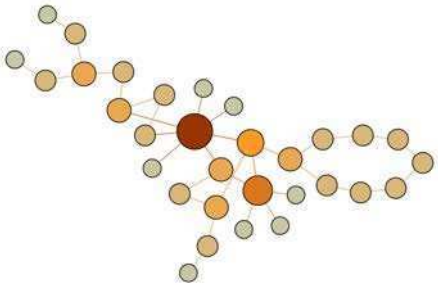
- The network *p2p-gnutella* listed is presented in [164, 128].

- *Power Grid* is a topology of the Western States Power Grid of the United States, documented in [199].

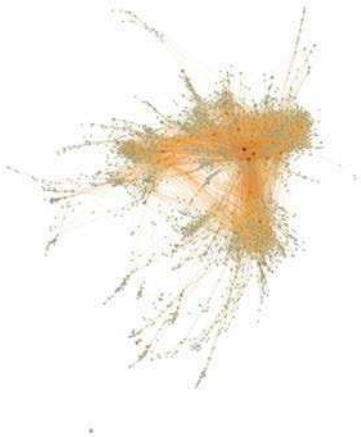
The detail of networks with their vertex and edge size is included in Table 4.6. For a sample of the visualization of some networks in this domain, refer to Figure 4.6.



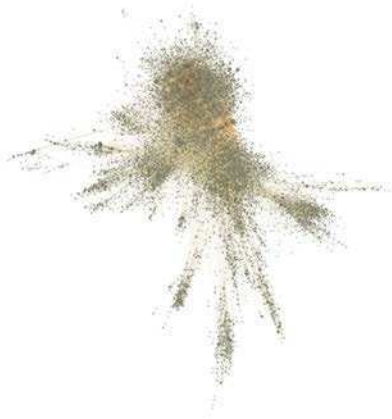
(a) Visualization of the "Top-Australia" network



(b) Visualization of the "Top-Brazil" network



(c) Visualization of the "openflights" network



(d) Visualization of the "Internet Routers" network

Figure 4.6: Visualization of four different networks from those selected for the Infrastructure domain

4.2.7 Metabolic

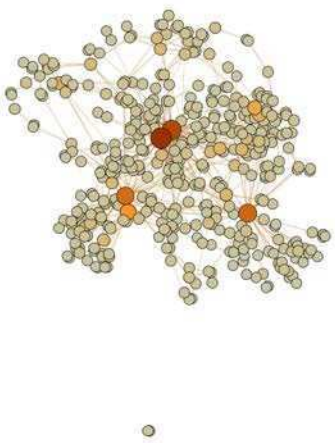
The domain labeled metabolic in this work, represents a collection of "metabolic networks" from different organisms. The vertices in these structures are the different substrates that take place in the metabolic reactions of each organism presented. The edges are determined by the existance of a reaction between the given substrates.

The list of networks used is shown on Table 4.7. Since for this domain we have available a set of networks with a unique and consistent origin, it presents itself as a good candidate for good classification and differentiation from the other domains, and we expect our experiments to give better results for these domains than for other with a more mixed set of networks, such as is the case for collaboration (p.75) and communication (p.80).

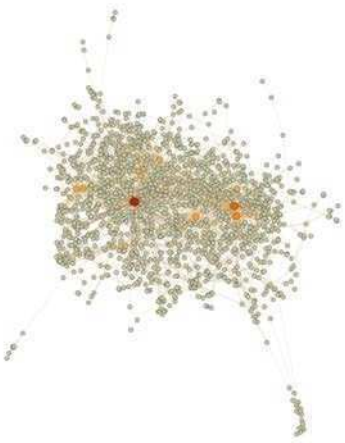
Table 4.7: Metabolic Networks

Graph	$\|V\|$	$\|E\|$	Graph	$\|V\|$	$\|E\|$
C. pneumoniae	387	792	S. pyogenes	1051	2577
B. burgdorferi	409	880	M. leprae	1055	2515
M. pneumoniae	411	936	N. gonorrhoeae	1055	2568
C. trachomatis	447	941	A. aeolicus	1057	2527
R. prowazekii	456	1014	S. pneumoniae	1077	2629
M. genitalium	473	1060	M. jannaschii	1082	2589
T. pallidum	485	1117	M. thermoautotro phicum	1112	2705
A. pernix	490	1163	C. elegans	1173	2864
O. sativa	665	1514	A. fulgidus	1268	3011
A. thaliana	694	1593	C. acetobutylicum	1305	3202
P. furiosus	751	1768	H. influenzae	1424	3519
P. horikoshii	767	1796	Synechocystis sp.	1430	3528
T. maritima	830	1980	Y. pestis	1453	3469
E. nidulans	916	2176	S. cerevisiae	1511	3833
C. tepidum	918	2159	M. tuberculosis	1520	3685
C. jejuni	946	2257	R. capsulatus	1762	4296
H. pylori	949	2325	P. aeruginosa	1971	4851
N. meningitidis	981	2393	B. subtilis	2217	5535
A. actinomycetem comitans	993	2368	E. coli	2275	5763
E. faecalis	1004	2462	D. radiodurans	2280	5681
P. gingivalis	1010	2348	S. typhi	2361	5959
M. bovis	1043	2468			

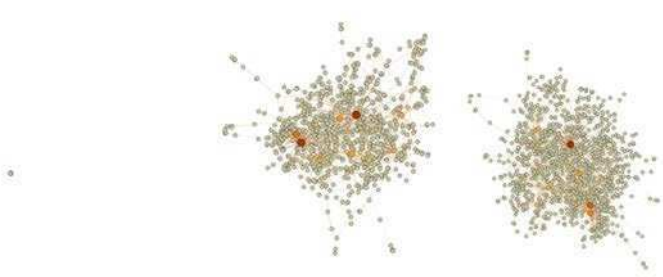
Table 4.7 contains all of the organisms selected, and indicates for each network the number of vertices and edges involved. Then, in Figure 4.7 we have chosen, as in previous domains, four networks to offer a visualization that allows a quick contract of their structures.



(a) Visualization of the "M. pneumoniae" network



(b) Visualization of the "S. cerevisiae" network



(c) Visualization of the "P. horikoshii" network

(d) Visualization of the "M. leprae" network

Figure 4.7: Visualization of four different networks from those selected for the Metabolic domain

4.2.8 Reaction Dependency Graph

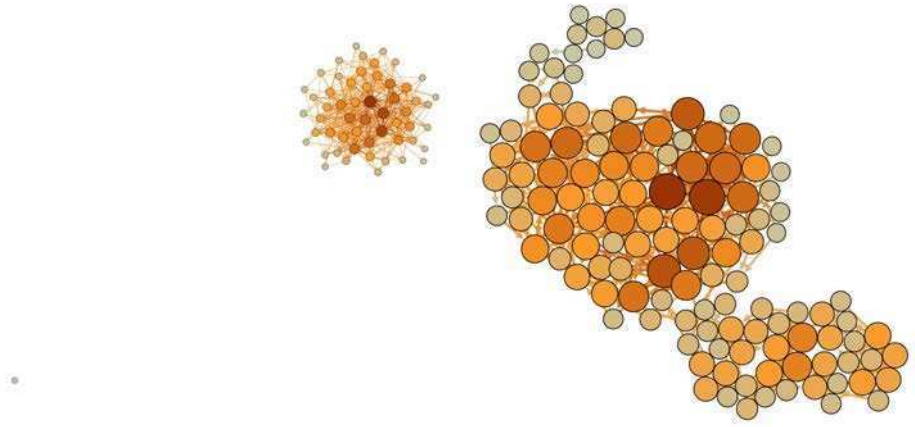
A **Reaction Dependency Graph** (RDG) is built from the relation between reactions in a biochemical model. These models imply an analysis of molecular species and the reactions between them. If a reaction R_i affects the propensity of another reaction R_j from occurring, an RDG will include vertices for R_i and R_j, as well as a directed edge (R_i, R_j).

Table 4.8: Reaction Dependency Graph Networks

Graph	$\|V\|$	$\|E\|$	Graph	$\|V\|$	$\|E\|$
BIOMD0000000103-RDG	42	348	BIOMD0000000056-RDG	94	424
BIOMD0000000139-RDG	45	318	BIOMD0000000175-RDG	102	921
BIOMD0000000303-RDG	45	345	BIOMD0000000223-RDG	109	677
BIOMD0000000333-RDG	45	333	BIOMD0000000244-RDG	109	347
BIOMD0000000147-RDG	51	355	BIOMD0000000338-RDG	115	452
BIOMD0000000256-RDG	56	574	BIOMD0000000339-RDG	115	452
BIOMD0000000328-RDG	58	588	BIOMD0000000340-RDG	116	457
BIOMD0000000268-RDG	61	310	BIOMD0000000152-RDG	120	1162
BIOMD0000000230-RDG	64	613	BIOMD0000000019-RDG	125	801
BIOMD0000000356-RDG	64	395	BIOMD0000000109-RDG	138	1570
BIOMD0000000334-RDG	66	591	BIOMD0000000049-RDG	150	1194
BIOMD0000000332-RDG	69	645	BIOMD0000000153-RDG	152	1734
BIOMD0000000250-RDG	78	334	BIOMD0000000014-RDG	300	7101
BIOMD0000000344-RDG	80	701	BIOMD0000000183-RDG	352	36672
BIOMD0000000326-RDG	91	582	BIOMD0000000255-RDG	827	25337

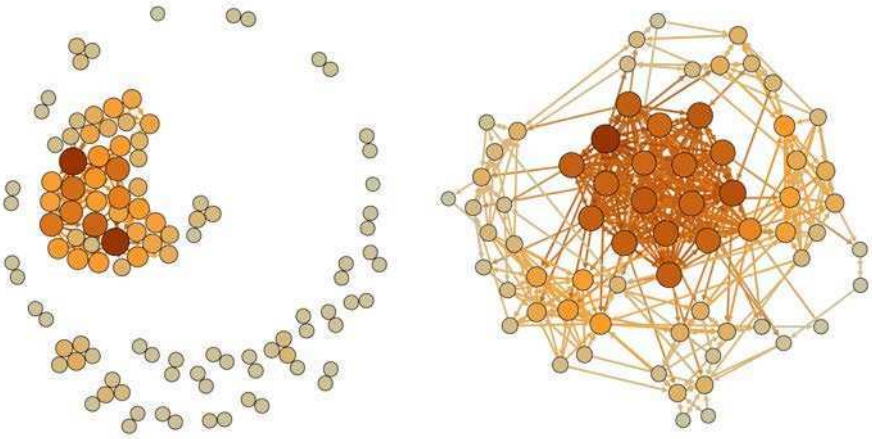
The set considered in this domain is built by taking 30 models obtained from [150], which have been processed and studied in detail in [169].

The number of vertices and edges for the networks is displayed in Table 4.8, and the visualization of reference structures is included in Figure 4.8.



(a) Visualization of the "BIOMD0000000256-RDG" network

(b) Visualization of the "BIOMD0000000019-RDG" network



(c) Visualization of the "BIOMD0000000244-RDG" network

(d) Visualization of the "BIOMD0000000332-RDG" network

Figure 4.8: Visualization of four different networks from those selected for the Reaction Dependency Graph domain

4.2.9 Social

The social domain presented here encompasses several different networks, which all share the property of representing a social organization, and the interrelation between its members. Such a general description most likely applies as well to some of the networks listed in previous domains. The general criteria applied was to relegate networks to this section only when (1) they were not explicitly compatible with previous definitions or (2) the original sources listed them in the "social" category, since analyzing the correctness of human-made classifications is part of our experimental objectives.

Most of the structures will of course have vertices to represent people, and edges when there is a connection in the corresponding social group represented.

Table 4.9: Social Networks

Graph	$\|V\|$	$\|E\|$
tribes	14	24
firm-hi-tech	33	147
Karate Club	34	78
Les Miserables	77	254
Jazz	198	2742
School 1	236	5899
School 2	238	5539
wiki-Vote	889	2914
EuroSiS	1285	7524
hamsterster	2426	16630

Graph	$\|V\|$	$\|E\|$
Twitter	3656	188712
advogato	5358	26788
wiki-elec	7118	103738
anybeat	12645	67053
gplus	23628	39242
epinions	26588	100120
sign-Slashdot081106	28598	78795
sign-Slashdot090216	29118	81134
sign-Slashdot090221	29131	81681

- *Karate Club* is the classical network of friendships between members of a karate club [209].

- *tribes* presents the result of recording friendship between tribes in New Guinea, as described in [160, 164].

- *Les Miserables* is another classical social network, where edges represent coappearance of characters in the novel Les Miserables [109].

- *School 1* and *School 2* both present edges according to face to face interactions in a school, during the first and second days of a study,

available at http://www.sociopatterns.org.

- *Jazz* is a network representing co-performances between jazz musicians [87].

- *advogato*[127], *anybeat*[76], *gplus* and *hamsterster*[95] are social network built using social relationships declared in the different sites. All of them can be obtained from [164].

- *epinions* has a structure built using trust relationships from Epinions.com [126, 164].

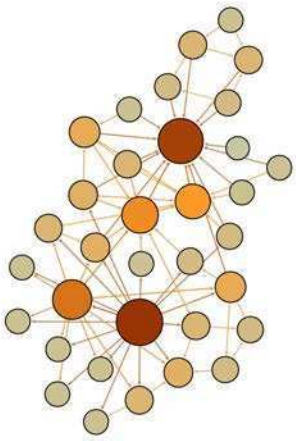
- *wiki-elec* and *wiki-Vote* represent information from administrator elections in Wikipedia, and accumulated voting information, respectively. More information can be found in [114, 164].

- *EuroSiS* is based on the relations between Science in Society actors on the Web of 12 European countries [83].

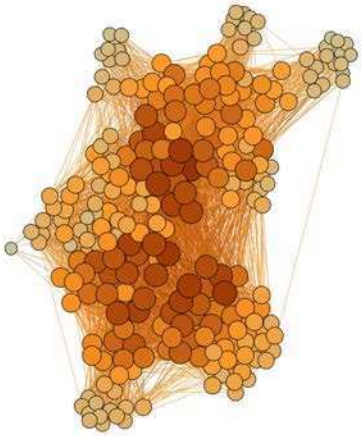
- *Twitter* is a collection of various Twitter mentions and retweets of some part of the Twitter network, provided by the statistics account https://twitter.com/rankinfo.

- *firm-hi-tech*, *sign-Slashdot081106*, *sign-Slashdot090216* and *sign-Slashdot090221* are additional networks presented in [164] in its "social" category.

All of the networks selected are listed in Table 4.9, along with the number of vertices and edges they present. In Figure 4.9, four of the selected structures are displayed to give a visualization example for this domain.

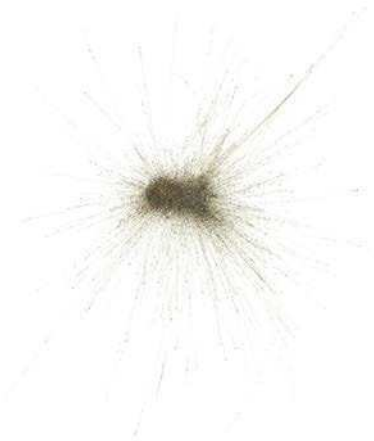


(a) Visualization of the "Karate Club" network



(b) Visualization of the "School 2" network

(c) Visualization of the "hamsterster" network



(d) Visualization of the "epinions" network

Figure 4.9: Visualization of four different networks from those selected for the Social domain

4.2.10 Species Network Graph

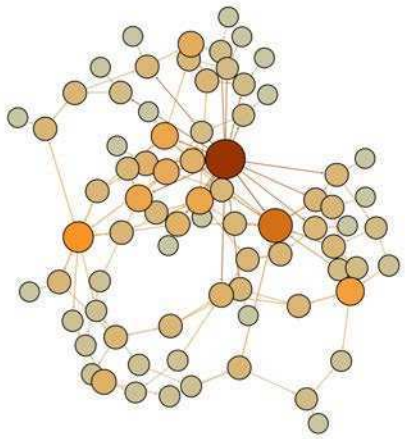
In contrast with the previously mentioned Reaction Dependency Graph (p.87), a **Species Network Graph** (SNG) is built from the relation between molecular species in a biochemical model. These models imply an analysis of molecular species and the reactions between them. If the model includes a reaction where S_i is a species that behaves as the reactant and S_j is the product, an SNG will include vertices for S_i and S_j, as well as a directed edge (S_i, S_j).

Table 4.10: Species Network Graph Networks

Graph	$\|V\|$	$\|E\|$	Graph	$\|V\|$	$\|E\|$
BIOMD0000000328-SNG	18	29	BIOMD0000000183-SNG	67	536
BIOMD0000000103-SNG	21	60	BIOMD0000000338-SNG	67	62
BIOMD0000000230-SNG	26	85	BIOMD0000000339-SNG	67	62
BIOMD0000000147-SNG	27	61	BIOMD0000000340-SNG	68	63
BIOMD0000000256-SNG	29	89	BIOMD0000000326-SNG	73	142
BIOMD0000000139-SNG	30	55	BIOMD0000000334-SNG	73	130
BIOMD0000000356-SNG	32	68	BIOMD0000000153-SNG	75	256
BIOMD0000000303-SNG	41	62	BIOMD0000000244-SNG	77	59
BIOMD0000000268-SNG	46	82	BIOMD0000000332-SNG	77	137
BIOMD0000000250-SNG	53	70	BIOMD0000000223-SNG	81	157
BIOMD0000000333-SNG	54	89	BIOMD0000000014-SNG	86	560
BIOMD0000000344-SNG	54	163	BIOMD0000000019-SNG	95	225
BIOMD0000000109-SNG	55	186	BIOMD0000000049-SNG	97	257
BIOMD0000000056-SNG	56	80	BIOMD0000000175-SNG	118	226
BIOMD0000000152-SNG	64	202	BIOMD0000000255-SNG	504	1350

As before, we will consider for this domain a total of 30 models obtained from [150], which have been processed and studied in detail in [169].

The number of vertices and edges for the networks is displayed in Table 4.10, and the visualization of reference structures is included in Figure 4.10.



(a) Visualization of the "BIOMD0000000332-SNG" network

(b) Visualization of the "BIOMD0000000356-SNG" network

(c) Visualization of the "BIOMD0000000340-SNG" network

(d) Visualization of the "BIOMD0000000056-SNG" network

Figure 4.10: Visualization of four different networks from those selected for the Species Network Graph domain

4.2.11 Tweet

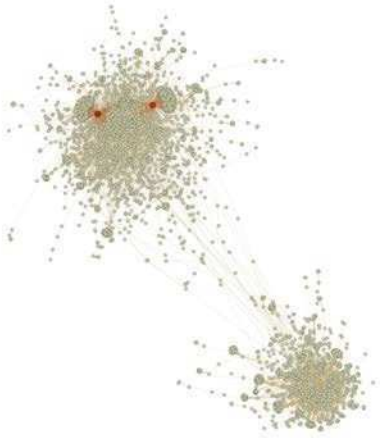
The last domain considered in this work is that of Tweeter retweets for different topics.

Using a selection of the networks referenced in [165, 166, 195], available at http://networkrepository.com, we have gathered an interesting set of structures. Each vertex represents a Twitter user u_i, while each edge (u_a, u_b) indicates that user u_a has retweeted user u_b on the hashtag involved.

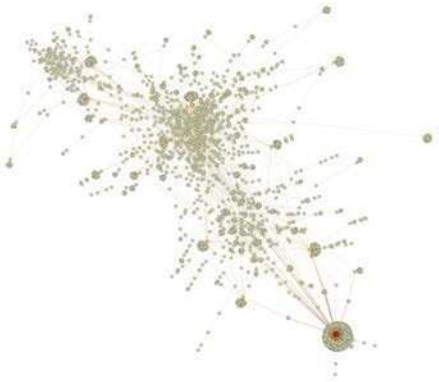
Table 4.11: Tweet Networks

Graph	$\|V\|$	$\|E\|$	Graph	$\|V\|$	$\|E\|$
NT-rt-retweet	96	117	NT-rt_p2	4902	6031
NT-rt-twitter-copen	761	1029	NT-rt_oman	4904	6236
NT-rt_assad	2139	2803	NT-rt_libya	5067	5550
NT-rt_voteonedirection	2280	2468	NT-rt_uae	5248	6394
NT-rt_damascus	3052	3881	NT-rt_dash	6288	7455
NT-rt_obama	3212	3427	NT-rt_saudi	7252	8094
NT-rt_occupy	3225	3964	NT-rt_ksa	7302	8125
NT-rt_occupywallstnyc	3609	3936	NT-rt_mittromney	7974	8597
NT-rt_tlot	3665	4481	NT-rt_onedirection	7987	8133
NT-rt_israel	3698	4175	NT-rt_gmanews	8373	8832
NT-rt_lebanon	3961	4441	NT-rt_http	8917	10318
NT-rt_alwefaq	4171	7123	NT-rt_justinbieber	9405	9615
NT-rt_islam	4497	4616	NT-rt_barackobama	9631	9826
NT-rt_tcot	4547	5516	NT-rt_lolgop	9765	10324
NT-rt_bahrain	4676	8007	NT-rt-pol	18470	48365
NT-rt_gop	4687	5536			

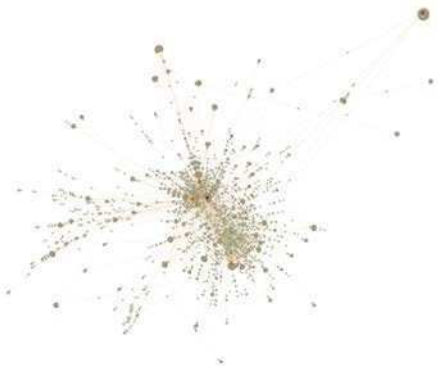
The complete set of networks selected for this domain is presented in Table 4.11, in ascending order of number of vertices. Figure 4.11 includes a visualization of four representative structures.



(a) Visualization of the "NT-rt_alwefaq" network



(b) Visualization of the "NT-rt_damascus" network



(c) Visualization of the "NT-rt_uae" network

(d) Visualization of the "NT-rt_http" network

Figure 4.11: Visualization of four different networks from those selected for the Tweet domain

4.3 A selection of measures

This section details the measures that will be considered in the rest of this work. They were mainly chosen by their simplicity of calculation, consistent implementation across different graph tools, and low correlation between them.

After completing the research of all the complexity measures described in chapter 2, we could see that many of them are combinations of others, which can lead to a high correlation. Since our aim will be to classify networks by means of their complexity, and this could require more or less detail depending on the dataset considered, we thought it would be better to have at our disposal a mix of many simple measures instead of few complex measures that inherently contain some of the simpler ones.

The following measures were chosen to be analyzed, by running them against all of the networks from the previously presented domains:

- $|V|$ (p. 19)
- $|E|$ (p. 19)
- Maximum Degree (p. 20)
- Diameter (p. 28)
- PathLength (p. 22)
- Closeness Centrality (p. 15)
- Betweenness Centrality (p. 25)
- Eigenvector Centrality (p. 32)
- Offdiagonal Complexity (p. 31)
- Clustering Coefficient (p. 29)
- Global Efficiency (p. 22)
- PageRank (p. 32)
- Codelength (p. 40)

For those measures where the value can be obtained for each vertex, we propose a series of derived measurements:

- Average

- Minimum
- Maximum
- 1^{st} Quartile
- 2^{nd} Quartile (Median)
- 3^{rd} Quartile
- Inter-Quartile Range

4.3.1 Correlation

To ensure that these measures are independent enough between them, we present now a correlation analysis. For those measures where the value depends on each vertex, we consider only the average value for all vertices in the network. The values presented are computed considering Pearson correlation[157].

First we will review the correlation in each domain, and then compare how they relate to the overall correlation. A color code has been chosen to help identify quickly the extreme values. The diagonal in the tables will be indicated with a teal colored cell, correlations that present an absolute value between 0.5 and 0.75 will be shown in yellow, and correlations above that range will be displayed in red. The abbreviations used in the following figures is presented in Table 4.12.

Table 4.12: Abbreviatons for the Complexity Measures in the Correlation results

$\|V\|$ → Number of Vertices	$\|E\|$ → Number of edges
CL → Codelength	Diam → Diameter
PL → PathLength	Betw → Betweenness Centrality
Close → Closeness Centrality	GE → Global Efficiency
MD → Maximum Degree	OdC → Offdiagonal Complexity
Clust → Clustering Coefficient	Eigen → Eigenvector Centrality
PR → PageRank	

Figure 4.12 is the first of our correlation tables, presenting the distribution in the Assorted Biological domain. We can observe the reasonable correlation between number of vertices and edges. The values for PageRank present a much lower relation with the other complexity measures. The correlation of Offdiagonal Complexity with Maximum Degree is expected, but it is interesting to observe the strong inverse correlation to both PageRank and Eigenvector Centrality. There is also a complete correlation between Eigenvector Centrality and PageRank.

The subset of Code networks present a strong correlation between various complexity measures, as show in Figure 4.13. Despite a lower relation of number of vertices with the number of edges, they both interact strongly with Betweenness and Closeness Centrality. They also share a high correlation with PathLength and Diameter. The low correlation between Eigenvector Centrality and PageRank is the lowest observed so far.

In Figure 4.14 we present correlations for the Collaboration domain. It shows a higher value for the relation of number of vertices and vertices to Diameter and PathLength, while a lower value for that of the first two with Betweenness and Closeness Centrality.

Figure 4.15 includes the correlations for the Combinatorial DNA Library Synthesis domain. Apart from the unsurprising correlation between the number of edges and vertices, we can observe an almost complete correlation between them both and Global Efficiency. We can also observe a very strong correlation between PathLength (and Betweenness Centrality)

	|V|	|E|	CL	Diam	PL	Betw	Close	GE	MD	OdC	Clust	Eigen	PR
|V|	1,00	0,99	0,69	0,53	0,32	0,22	0,28	0,11	0,21	0,30	-0,43	-0,33	-0,31
|E|	0,99	1,00	0,76	0,49	0,26	0,16	0,28	-0,01	0,30	0,35	-0,38	-0,30	-0,28
CL	0,69	0,76	1,00	0,30	-0,08	-0,16	-0,02	-0,12	0,59	0,72	-0,16	-0,44	-0,45
Diam	0,53	0,49	0,30	1,00	0,88	0,75	0,65	-0,15	-0,01	0,18	-0,41	-0,30	-0,30
PL	0,32	0,26	-0,08	0,88	1,00	0,85	0,66	-0,26	-0,13	-0,04	-0,19	-0,10	-0,10
Betw	0,22	0,16	-0,16	0,75	0,85	1,00	0,73	-0,27	0,04	0,00	-0,10	-0,24	-0,24
Close	0,28	0,28	-0,02	0,65	0,66	0,73	1,00	-0,43	-0,06	-0,27	-0,50	0,26	0,27
GE	0,11	-0,01	-0,12	-0,15	-0,26	-0,27	-0,43	1,00	-0,33	-0,08	-0,28	-0,24	-0,22
MD	0,21	0,30	0,59	-0,01	-0,13	0,04	-0,06	-0,33	1,00	0,83	0,46	-0,47	-0,49
OdC	0,30	0,35	0,72	0,18	-0,04	0,00	-0,27	-0,08	0,83	1,00	0,44	-0,80	-0,82
Clust	-0,43	-0,38	-0,16	-0,41	-0,19	-0,10	-0,50	-0,28	0,46	0,44	1,00	-0,35	-0,35
Eigen	-0,33	-0,30	-0,44	-0,30	-0,10	-0,24	0,26	-0,24	-0,47	-0,80	-0,35	1,00	1,00
PR	-0,31	-0,28	-0,45	-0,30	-0,10	-0,24	0,27	-0,22	-0,49	-0,82	-0,35	1,00	1,00

Figure 4.12: Correlation between the studied measures for all networks in the Assorted Biological domain

	|V|	|E|	CL	Diam	PL	Betw	Close	GE	MD	OdC	Clust	Eigen	PR
|V|	1,00	0,82	0,70	0,64	0,68	0,99	0,80	-0,23	0,76	0,47	-0,02	-0,54	-0,62
|E|	0,82	1,00	0,80	0,31	0,40	0,77	0,57	-0,32	0,89	0,61	0,06	-0,30	-0,50
CL	0,70	0,80	1,00	0,32	0,38	0,72	0,61	-0,61	0,91	0,85	0,29	0,07	-0,33
Diam	0,64	0,31	0,32	1,00	0,95	0,73	0,89	-0,47	0,24	0,19	0,27	-0,35	-0,46
PL	0,68	0,40	0,38	0,95	1,00	0,76	0,94	-0,55	0,36	0,27	0,36	-0,28	-0,42
Betw	0,99	0,77	0,72	0,73	0,76	1,00	0,87	-0,34	0,73	0,46	0,04	-0,47	-0,56
Close	0,80	0,57	0,61	0,89	0,94	0,87	1,00	-0,60	0,53	0,42	0,31	-0,24	-0,38
GE	-0,23	-0,32	-0,61	-0,47	-0,55	-0,34	-0,60	1,00	-0,44	-0,57	-0,61	-0,44	-0,11
MD	0,76	0,89	0,91	0,24	0,36	0,73	0,53	-0,44	1,00	0,86	0,30	-0,05	-0,52
OdC	0,47	0,61	0,85	0,19	0,27	0,46	0,42	-0,57	0,86	1,00	0,64	0,34	-0,45
Clust	-0,02	0,06	0,29	0,27	0,36	0,04	0,31	-0,61	0,30	0,64	1,00	0,67	-0,23
Eigen	-0,54	-0,30	0,07	-0,35	-0,28	-0,47	-0,24	-0,44	-0,05	0,34	0,67	1,00	0,45
PR	-0,62	-0,50	-0,33	-0,46	-0,42	-0,56	-0,38	-0,11	-0,52	-0,45	-0,23	0,45	1,00

Figure 4.13: Correlation between the studied measures for all networks in the Code domain

to the network Diameter. This figure presents as well a negative correlation between the values obtained for PageRank and several of the other complexity measures.

For the networks in the Communication domain, Figure 4.16 presents

	|V|	|E|	CL	Diam	PL	Betw	Close	GE	MD	OdC	Clust	Eigen	PR
|V|	1,00	0,78	0,71	0,83	0,79	0,47	0,47	-0,14	0,49	0,40	0,44	-0,49	-0,48
|E|	0,78	1,00	0,71	0,59	0,63	0,87	0,66	-0,31	0,78	0,46	0,72	-0,35	-0,37
CL	0,71	0,71	1,00	0,57	0,66	0,48	0,75	-0,36	0,75	0,76	0,45	-0,55	-0,52
Diam	0,83	0,59	0,57	1,00	0,89	0,30	0,39	-0,16	0,25	0,22	0,35	-0,37	-0,52
PL	0,79	0,63	0,66	0,89	1,00	0,44	0,66	-0,28	0,41	0,42	0,42	-0,45	-0,55
Betw	0,47	0,87	0,48	0,30	0,44	1,00	0,62	-0,26	0,81	0,36	0,77	-0,22	-0,26
Close	0,47	0,66	0,75	0,39	0,66	0,62	1,00	-0,57	0,66	0,51	0,65	-0,33	-0,27
GE	-0,14	-0,31	-0,36	-0,16	-0,28	-0,26	-0,57	1,00	-0,13	0,01	-0,65	-0,19	-0,38
MD	0,49	0,78	0,75	0,25	0,41	0,81	0,66	-0,13	1,00	0,75	0,58	-0,50	-0,51
OdC	0,40	0,46	0,76	0,22	0,42	0,36	0,51	0,01	0,75	1,00	0,18	-0,75	-0,67
Clust	0,44	0,72	0,45	0,35	0,42	0,77	0,65	-0,65	0,58	0,18	1,00	-0,20	-0,07
Eigen	-0,49	-0,35	-0,55	-0,37	-0,45	-0,22	-0,33	-0,19	-0,50	-0,75	-0,20	1,00	0,79
PR	-0,48	-0,37	-0,52	-0,52	-0,55	-0,26	-0,27	-0,38	-0,51	-0,67	-0,07	0,79	1,00

Figure 4.14: Correlation between the studied measures for all networks in the Collaboration domain

	|V|	|E|	CL	Diam	PL	Betw	Close	GE	MD	OdC	Clust	Eigen	PR
|V|	1,00	1,00	0,72	0,37	0,31	0,19	-0,11	0,97	0,76	0,35	0,00	0,34	-0,58
|E|	1,00	1,00	0,73	0,36	0,31	0,19	-0,13	0,96	0,76	0,33	0,00	0,37	-0,59
CL	0,72	0,73	1,00	0,50	0,57	0,41	-0,14	0,60	0,69	0,38	0,00	0,77	-0,88
Diam	0,37	0,36	0,50	1,00	0,94	0,87	0,72	0,22	0,43	0,64	0,00	0,04	-0,58
PL	0,31	0,31	0,57	0,94	1,00	0,93	0,70	0,13	0,44	0,50	0,00	0,21	-0,62
Betw	0,19	0,19	0,41	0,87	0,93	1,00	0,70	-0,02	0,25	0,36	0,00	0,15	-0,39
Close	-0,11	-0,13	-0,14	0,72	0,70	0,70	1,00	-0,20	0,07	0,34	0,00	-0,49	-0,03
GE	0,97	0,96	0,60	0,22	0,13	-0,02	-0,20	1,00	0,73	0,36	0,00	0,23	-0,51
MD	0,76	0,76	0,69	0,43	0,44	0,25	0,07	0,73	1,00	0,39	0,00	0,33	-0,69
OdC	0,35	0,33	0,38	0,64	0,50	0,36	0,34	0,36	0,39	1,00	0,00	-0,19	-0,45
Clust	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	1,00	0,00	0,00
Eigen	0,34	0,37	0,77	0,04	0,21	0,15	-0,49	0,23	0,33	-0,19	0,00	1,00	-0,64
PR	-0,58	-0,59	-0,88	-0,58	-0,62	-0,39	-0,03	-0,51	-0,69	-0,45	0,00	-0,64	1,00

Figure 4.15: Correlation between the studied measures for all networks in the Combinatorial DNA Library Synthesis domain

	|V|	|E|	CL	Diam	PL	Betw	Close	GE	MD	OdC	Clust	Eigen	PR
|V|	1,00	0,81	0,28	0,24	0,98	0,83	0,29	-0,10	0,97	0,65	-0,25	-0,31	-0,17
|E|	0,81	1,00	0,51	0,37	0,77	0,99	0,50	-0,15	0,91	0,72	0,08	-0,37	-0,21
CL	0,28	0,51	1,00	0,73	0,33	0,51	0,64	-0,34	0,40	0,63	-0,24	-0,58	-0,60
Diam	0,24	0,37	0,73	1,00	0,33	0,34	0,46	-0,11	0,30	0,43	-0,52	-0,82	-0,51
PL	0,98	0,77	0,33	0,33	1,00	0,79	0,32	-0,15	0,95	0,59	-0,33	-0,38	-0,23
Betw	0,83	0,99	0,51	0,34	0,79	1,00	0,45	-0,17	0,93	0,69	0,06	-0,34	-0,19
Close	0,29	0,50	0,64	0,46	0,32	0,45	1,00	-0,28	0,34	0,36	-0,01	-0,39	-0,27
GE	-0,10	-0,15	-0,34	-0,11	-0,15	-0,17	-0,28	1,00	-0,14	0,03	-0,36	-0,28	-0,15
MD	0,97	0,91	0,40	0,30	0,95	0,93	0,34	-0,14	1,00	0,71	-0,15	-0,34	-0,20
OdC	0,65	0,72	0,63	0,43	0,59	0,69	0,36	0,03	0,71	1,00	-0,17	-0,41	-0,55
Clust	-0,25	0,08	-0,24	-0,52	-0,33	0,06	-0,01	-0,36	-0,15	-0,17	1,00	0,75	0,46
Eigen	-0,31	-0,37	-0,58	-0,82	-0,38	-0,34	-0,39	-0,28	-0,34	-0,41	0,75	1,00	0,68
PR	-0,17	-0,21	-0,60	-0,51	-0,23	-0,19	-0,27	-0,15	-0,20	-0,55	0,46	0,68	1,00

Figure 4.16: Correlation between the studied measures for all networks in the Communication domain

a set of values that is similar in many ways to that observed for the Code domain. Maximum Degree appears now strongly correlated to number of vertices and edges, but also to PathLength and Betweenness Centrality.

	|V|	|E|	CL	Diam	PL	Betw	Close	GE	MD	OdC	Clust	Eigen	PR
|V|	1,00	0,96	0,76	0,79	0,73	0,81	0,65	0,09	0,58	0,44	-0,13	-0,53	-0,33
|E|	0,96	1,00	0,84	0,84	0,77	0,83	0,73	0,03	0,55	0,52	-0,02	-0,58	-0,37
CL	0,76	0,84	1,00	0,83	0,79	0,61	0,80	0,09	0,59	0,83	0,27	-0,77	-0,62
Diam	0,79	0,84	0,83	1,00	0,97	0,82	0,93	0,15	0,28	0,44	-0,07	-0,66	-0,52
PL	0,73	0,77	0,79	0,97	1,00	0,80	0,95	-0,00	0,21	0,41	-0,07	-0,60	-0,51
Betw	0,81	0,83	0,61	0,82	0,80	1,00	0,75	-0,10	0,03	0,14	-0,08	-0,30	-0,22
Close	0,65	0,73	0,80	0,93	0,95	0,75	1,00	-0,04	0,21	0,47	0,11	-0,57	-0,44
GE	0,09	0,03	0,09	0,15	-0,00	-0,10	-0,04	1,00	0,16	0,07	-0,12	-0,33	-0,23
MD	0,58	0,55	0,59	0,28	0,21	0,03	0,21	0,16	1,00	0,73	0,08	-0,56	-0,34
OdC	0,44	0,52	0,83	0,44	0,41	0,14	0,47	0,07	0,73	1,00	0,52	-0,67	-0,60
Clust	-0,13	-0,02	0,27	-0,07	-0,07	-0,08	0,11	-0,12	0,08	0,52	1,00	-0,05	0,10
Eigen	-0,53	-0,58	-0,77	-0,66	-0,60	-0,30	-0,57	-0,33	-0,56	-0,67	-0,05	1,00	0,68
PR	-0,33	-0,37	-0,62	-0,52	-0,51	-0,22	-0,44	-0,23	-0,34	-0,60	0,10	0,68	1,00

Figure 4.17: Correlation between the studied measures for all networks in the Infrastructure domain

Infrastructure networks show a very strong correlation (Figure 4.17) between half of the complexity measures presented: number of vertices, number of edges, Codelength, Diameter, PathLength, Betweenness Centrality and Closeness Centrality. The other values observed and nowhere as high as these, which might indicate that the physical reality on which the networks are based might affect the relation between the different properties.

	|V|	|E|	CL	Diam	PL	Betw	Close	GE	MD	OdC	Clust	Eigen	PR
|V|	1,00	1,00	0,90	0,02	-0,36	1,00	-0,14	-0,42	0,99	0,89	0,00	-0,71	-0,86
|E|	1,00	1,00	0,91	0,02	-0,38	0,99	-0,15	-0,44	0,99	0,89	0,00	-0,71	-0,85
CL	0,90	0,91	1,00	0,13	-0,44	0,91	-0,12	-0,67	0,86	0,96	0,00	-0,79	-0,97
Diam	0,02	0,02	0,13	1,00	0,50	0,07	0,58	0,08	-0,01	0,16	0,00	-0,16	-0,23
PL	-0,36	-0,38	-0,44	0,50	1,00	-0,30	0,91	0,65	-0,39	-0,38	0,00	0,37	0,33
Betw	1,00	0,99	0,91	0,07	-0,30	1,00	-0,06	-0,41	0,98	0,90	0,00	-0,71	-0,86
Close	-0,14	-0,15	-0,12	0,58	0,91	-0,06	1,00	0,30	-0,19	-0,09	0,00	0,21	0,04
GE	-0,42	-0,44	-0,67	0,08	0,65	-0,41	0,30	1,00	-0,40	-0,55	0,00	0,38	0,56
MD	0,99	0,99	0,86	-0,01	-0,39	0,98	-0,19	-0,40	1,00	0,84	0,00	-0,65	-0,79
OdC	0,89	0,89	0,96	0,16	-0,38	0,90	-0,09	-0,55	0,84	1,00	0,00	-0,78	-0,96
Clust	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	1,00	0,00	0,00
Eigen	-0,71	-0,71	-0,79	-0,16	0,37	-0,71	0,21	0,38	-0,65	-0,78	0,00	1,00	0,84
PR	-0,86	-0,85	-0,97	-0,23	0,33	-0,86	0,04	0,56	-0,79	-0,96	0,00	0,84	1,00

Figure 4.18: Correlation between the studied measures for all networks in the Metabolic domain

The metabolic networks studied have a correlation distribution for the studied measures that is strikingly different from that observed so far. In Figure 4.18 we can observe a very strong correlation between number of vertices to: number of edges, CodeLength, Betweenness Centrality, Maximum Degree, OffDiagonal Complexity, and a negative correlation to PageRank. Maximum Degree and OffDiagonal Complexity also present a high correla-

tion to various other measures. PageRank also present a negative correlation to more than half of the other measures.

	|V|	|E|	CL	Diam	PL	Betw	Close	GE	MD	OdC	Clust	Eigen	PR
|V|	1,00	0,78	0,53	0,31	0,21	0,93	0,24	-0,07	0,62	0,33	0,38	-0,20	-0,65
|E|	0,78	1,00	0,66	-0,00	-0,07	0,60	0,03	-0,19	0,96	0,34	0,49	0,05	-0,50
CL	0,53	0,66	1,00	-0,05	-0,04	0,43	0,26	-0,69	0,74	0,85	0,09	0,25	-0,42
Diam	0,31	-0,00	-0,05	1,00	0,95	0,51	0,83	0,02	-0,13	-0,09	-0,21	-0,43	-0,27
PL	0,21	-0,07	-0,04	0,95	1,00	0,41	0,89	-0,01	-0,17	-0,04	-0,28	-0,46	-0,28
Betw	0,93	0,60	0,43	0,51	0,41	1,00	0,47	-0,19	0,39	0,28	0,29	-0,17	-0,47
Close	0,24	0,03	0,26	0,83	0,89	0,47	1,00	-0,42	-0,03	0,24	-0,20	-0,21	-0,18
GE	-0,07	-0,19	-0,69	0,02	-0,01	-0,19	-0,42	1,00	-0,27	-0,60	0,01	-0,59	-0,24
MD	0,62	0,96	0,74	-0,13	-0,17	0,39	-0,03	-0,27	1,00	0,44	0,43	0,13	-0,46
OdC	0,33	0,34	0,85	-0,09	-0,04	0,28	0,24	-0,60	0,44	1,00	-0,05	0,13	-0,33
Clust	0,38	0,49	0,09	-0,21	-0,28	0,29	-0,20	0,01	0,43	-0,05	1,00	0,18	-0,03
Eigen	-0,20	0,05	0,25	-0,43	-0,46	-0,17	-0,21	-0,59	0,13	0,13	0,18	1,00	0,57
PR	-0,65	-0,50	-0,42	-0,27	-0,28	-0,47	-0,18	-0,24	-0,46	-0,33	-0,03	0,57	1,00

Figure 4.19: Correlation between the studied measures for all networks in the Reaction Dependency Graph domain

In the case of Reaction Dependency Graphs, we display in Figure 4.19 the results of computing the correlation for the different measures. Except for a few high values such as Diameter to PathLength or Number of edges to maximum degree, it is one of the least correlated domains of those reviewed so far.

	|V|	|E|	CL	Diam	PL	Betw	Close	GE	MD	OdC	Clust	Eigen	PR
|V|	1,00	0,52	0,70	0,74	0,78	0,78	-0,05	0,33	0,21	0,55	-0,75	-0,56	-0,38
|E|	0,52	1,00	0,69	0,57	0,52	0,57	0,41	-0,00	0,25	0,76	-0,52	-0,46	-0,40
CL	0,70	0,69	1,00	0,83	0,77	0,70	0,33	0,01	0,29	0,86	-0,70	-0,66	-0,74
Diam	0,74	0,57	0,83	1,00	0,93	0,65	0,48	0,01	0,09	0,66	-0,64	-0,65	-0,66
PL	0,78	0,52	0,77	0,93	1,00	0,75	0,32	0,07	0,04	0,60	-0,73	-0,61	-0,60
Betw	0,78	0,57	0,70	0,65	0,75	1,00	0,08	-0,15	0,33	0,53	-0,67	-0,51	-0,34
Close	-0,05	0,41	0,33	0,48	0,32	0,08	1,00	-0,50	-0,01	0,31	-0,03	-0,10	-0,38
GE	0,33	-0,00	0,01	0,01	0,07	-0,15	-0,50	1,00	0,22	0,25	-0,14	-0,12	-0,13
MD	0,21	0,25	0,29	0,09	0,04	0,33	-0,01	0,22	1,00	0,53	-0,11	-0,29	-0,22
OdC	0,55	0,76	0,86	0,66	0,60	0,53	0,31	0,25	0,53	1,00	-0,46	-0,59	-0,79
Clust	-0,75	-0,52	-0,70	-0,64	-0,73	-0,67	-0,03	-0,14	-0,11	-0,46	1,00	0,71	0,24
Eigen	-0,56	-0,46	-0,66	-0,65	-0,61	-0,51	-0,10	-0,12	-0,29	-0,59	0,71	1,00	0,53
PR	-0,38	-0,40	-0,74	-0,66	-0,60	-0,34	-0,38	-0,13	-0,22	-0,79	0,24	0,53	1,00

Figure 4.20: Correlation between the studied measures for all networks in the Social domain

Social networks present in Figure 4.20 another scenario that is different from those previously seen up to now. There is a strong correlation for the first half of the measures, similar to what was observed for the Infrastructure domain, but there is a distinct lack of correlation for Closeness Centrality, Global Efficiency and Maximum Degree to all of the others.

The domain of Species Network Graph has its correlation displayed in Figure 4.21. The most relevant relations are between Diameter, Path-

Length, Betweenness and Closeness Centrality. The other measures present in general a much lower value.

The last domain is that of Tweeter retweets. Figure 4.22 allows a glimpse at the relatively low correlation in the measures computed for these network set. Betweenness Centrality is highly correlated to number of vertices and edges, and also to CodeLength. These last three are also highly correlated between themselves. But the other interactions are remarkably low.

With all the correlations reviewed per domain, we proceed now to computing the correlation between the measurements for all the networks under study.

Figure 4.23 is not an average of the values previously presented, but the computation of the correlation considering all networks as a complete set.

In this final comparison, we observe a relatively low correlation between the chosen measures, except for number of vertices, number of edges, which are naturally involved in any other calculation.

	|V|	|E|	CL	Diam	PL	Betw	Close	GE	MD	OdC	Clust	Eigen	PR
|V|	1,00	0,90	0,50	0,12	0,03	0,02	-0,10	0,72	0,49	0,40	-0,16	-0,48	-0,52
|E|	0,90	1,00	0,70	0,05	-0,02	0,09	-0,04	0,48	0,75	0,50	0,18	-0,27	-0,46
CL	0,50	0,70	1,00	0,40	0,39	0,45	0,43	-0,16	0,62	0,82	0,37	0,09	-0,33
Diam	0,12	0,05	0,40	1,00	0,97	0,82	0,89	-0,25	-0,13	0,35	-0,24	-0,06	-0,45
PL	0,03	-0,02	0,39	0,97	1,00	0,85	0,93	-0,36	-0,15	0,37	-0,19	-0,02	-0,39
Betw	0,02	0,09	0,45	0,82	0,85	1,00	0,91	-0,37	-0,00	0,33	0,04	0,08	-0,38
Close	-0,10	-0,04	0,43	0,89	0,93	0,91	1,00	-0,50	-0,09	0,37	0,04	0,11	-0,28
GE	0,72	0,48	-0,16	-0,25	-0,36	-0,37	-0,50	1,00	0,13	-0,20	-0,40	-0,68	-0,40
MD	0,49	0,75	0,62	-0,13	-0,15	-0,00	-0,09	0,13	1,00	0,27	0,58	-0,16	-0,24
OdC	0,40	0,50	0,82	0,35	0,37	0,33	0,37	-0,20	0,27	1,00	0,29	0,05	-0,22
Clust	-0,16	0,18	0,37	-0,24	-0,19	0,04	0,04	-0,40	0,58	0,29	1,00	0,17	0,18
Eigen	-0,48	-0,27	0,09	-0,06	-0,02	0,08	0,11	-0,68	-0,16	0,05	0,17	1,00	0,60
PR	-0,52	-0,46	-0,33	-0,45	-0,39	-0,38	-0,28	-0,40	-0,24	-0,22	0,18	0,60	1,00

Figure 4.21: Correlation between the studied measures for all networks in the Species Network Graph domain

	|V|	|E|	CL	Diam	PL	Betw	Close	GE	MD	OdC	Clust	Eigen	PR
|V|	1,00	0,88	0,78	0,17	0,05	0,80	0,18	0,07	0,48	-0,10	-0,18	-0,15	-0,36
|E|	0,88	1,00	0,75	0,30	0,25	0,91	0,43	-0,03	0,13	0,19	-0,00	-0,16	-0,21
CL	0,78	0,75	1,00	0,36	0,36	0,77	0,28	-0,28	0,51	-0,18	0,17	-0,04	-0,36
Diam	0,17	0,30	0,36	1,00	0,92	0,35	0,45	-0,18	-0,22	0,45	0,03	-0,24	-0,24
PL	0,05	0,25	0,36	0,92	1,00	0,35	0,44	-0,29	-0,29	0,42	0,10	-0,15	-0,14
Betw	0,80	0,91	0,77	0,35	0,35	1,00	0,61	-0,30	0,23	-0,01	-0,01	-0,13	-0,11
Close	0,18	0,43	0,28	0,45	0,44	0,61	1,00	-0,47	-0,10	0,08	0,19	-0,25	0,25
GE	0,07	-0,03	-0,28	-0,18	-0,29	-0,30	-0,47	1,00	-0,31	0,41	-0,23	-0,22	-0,15
MD	0,48	0,13	0,51	-0,22	-0,29	0,23	-0,10	-0,31	1,00	-0,83	-0,08	0,13	-0,17
OdC	-0,10	0,19	-0,18	0,45	0,42	-0,01	0,08	0,41	-0,83	1,00	0,06	-0,32	-0,13
Clust	-0,18	-0,00	0,17	0,03	0,10	-0,01	0,19	-0,23	-0,08	0,06	1,00	-0,10	0,33
Eigen	-0,15	-0,16	-0,04	-0,24	-0,15	-0,13	-0,25	-0,22	0,13	-0,32	-0,10	1,00	0,05
PR	-0,36	-0,21	-0,36	-0,24	-0,14	-0,11	0,25	-0,15	-0,17	-0,13	0,33	0,05	1,00

Figure 4.22: Correlation between the studied measures for all networks in the Tweet domain

When reviewing the individual domain correlations, except for Eigenvector Centrality and PageRank, all other measures had a tendency to have a relatively small (< 0.75) positive correlation with each other, and negative against these two. The values were in some cases above 0.8. This could prove important from a classification perspective, since the correlation patterns for different kind of networks is, itself, different. Thus providing strong evidence that these preselected set of network features could form a set of informative descriptors for classifying networks automatically. In the general correlation analysis almost all of these influences seem to be smoothed up to the point in which it seems that there is almost no influence of measure over another.

Furthermore, when considering the highly correlated measures in this last table, they could be seen as candidates for a future reduction in the number of complexity measures, since they seem to share equivalent values. But since many of them were more correlated for specific domains, a relation between them could be instead added to the list of measures to experiment on. They could very well be used in devising a significant way to distinguish one domain from another, which is one of the main objectives of this work.

4.3.2 Real Time Complexity

Although Chapter 2 includes an estimated time complexity for all of the measures selected, the real computing times of an algorithm can vary greatly depending on the input parameters and the details of its implementation.

When computing the values for the complexity measures for all of the networks presented, the running time required to compute each of them was logged. We then considered the total time spent on each measure (the sum

	|V|	|E|	CL	Diam	PL	Betw	Close	GE	MD	OdC	Clust	Eigen	PR
|V|	1,00	0,75	0,31	0,13	0,54	0,81	0,01	0,04	0,46	0,36	-0,07	-0,17	-0,13
|E|	0,75	1,00	0,49	0,17	0,39	0,88	0,09	-0,03	0,38	0,48	0,11	-0,21	-0,17
CL	0,31	0,49	1,00	0,55	0,45	0,43	0,35	-0,06	0,30	0,73	-0,07	-0,33	-0,58
Diam	0,13	0,17	0,55	1,00	0,77	0,26	0,69	-0,06	0,01	0,37	-0,35	-0,44	-0,47
PL	0,54	0,39	0,45	0,77	1,00	0,54	0,61	-0,16	0,10	0,31	-0,32	-0,22	-0,39
Betw	0,81	0,88	0,43	0,26	0,54	1,00	0,22	-0,09	0,36	0,38	0,01	-0,21	-0,16
Close	0,01	0,09	0,35	0,69	0,61	0,22	1,00	-0,34	-0,18	0,13	-0,09	-0,20	-0,21
GE	0,04	-0,03	-0,06	-0,06	-0,16	-0,09	-0,34	1,00	0,05	0,20	-0,20	-0,17	-0,16
MD	0,46	0,38	0,30	0,01	0,10	0,36	-0,18	0,05	1,00	0,18	-0,11	-0,22	-0,19
OdC	0,36	0,48	0,73	0,37	0,31	0,38	0,13	0,20	0,18	1,00	0,07	-0,56	-0,49
Clust	-0,07	0,11	-0,07	-0,35	-0,32	0,01	-0,09	-0,20	-0,11	0,07	1,00	0,17	0,15
Eigen	-0,17	-0,21	-0,33	-0,44	-0,22	-0,21	-0,20	-0,17	-0,22	-0,56	0,17	1,00	0,31
PR	-0,13	-0,17	-0,58	-0,47	-0,39	-0,16	-0,21	-0,16	-0,19	-0,49	0,15	0,31	1,00

Figure 4.23: Correlation between the studied measures for all networks in all domains

Table 4.13: Relative Running Time

Measure	Theoretical Time	Observed Time
Maximum Degree	$\mathbf{O}(\|V\|)$	0.1%
Diameter	$\mathbf{O}(\|V\|^3)$	5.5%
PathLength	$\mathbf{O}(\|V\|^3)$	5.5%
Closeness Centrality	$\mathbf{O}(\|V\|^3)$	5.5%
Betweenness Centrality	$\mathbf{O}(\|V\|^3)$	5.5%
Eigenvector Centrality	$\mathbf{O}(\|V\| + \|E\|)$	3.7%
Offdiagonal Complexity	$\mathbf{O}(\|E\|)$	0.4%
Clustering Coefficient	$\mathbf{O}(\|V\|^3)$	1.5%
Global Efficiency	$\mathbf{O}(\|V\|^2)$	48.8%
PageRank	$\mathbf{O}(\|V\| + \|E\|)$	1.2%
Codelength	Polynomial	22.3%

of the time spent for all networks on that measure), related to the total required for all the calculations performed. This relative running time is presented in Table 4.13. For each of the measures chosen we can compare the theoretical time (in terms of time complexity), and the percentage of the total time running observed during the computations.

Even measures that theoretically present a greater time complexity than others have their real execution times with the opposite relation in some cases, as can be seen for the Global Efficiency, which has taken almost half of the running time for all the different measures. One of the possible reasons for this behavior is that different measures might make assumptions about the network structure, or be optimal in scenarios where computer memory is not a factor. After analyzing the usefulness of the measures in the following chapters, the most promising ones could be subjected to additional implementation improvements to ensure an overall increase in performance for the experiments proposed.

4.3.3 Statistics per Network Class

Since we are interested in finding the differentiating factors between networks domains, and have so far chosen a set of complexity measures and networks to apply them to, we focus now on understanding how the values for those measurements are distributed across the different domains.

This final section of the chapter will look at a series of charts that present the statistical distribution of the measurements observed. This will allow us to better know what results are to be expected from the algorithms

presented in Chapter 3.

The values computed for all networks have been normalized so that, regardless of domain, all values for each measure will be between 0 and 1. Furthermore, based on the measurements values, we have determined the minimum, maximum, median, first and third quartile of each data series, thus allowing the construction of a "boxplot" chart for each one of the complexity measures chosen.

Figures 4.24 and 4.25 present the size measures (number of vertices and number of edges). The extreme values that can be observed for some categories evidence the need to carefully detect outliers when preparing the experiments, which will be shown in following chapters.

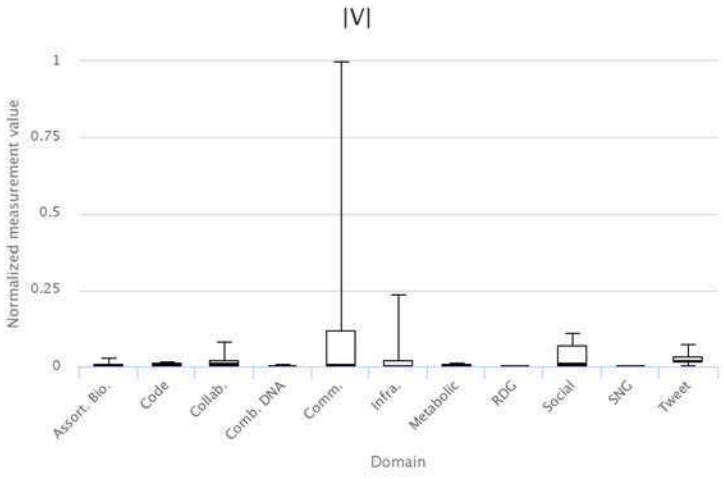


Figure 4.24: Statistical distribution of the observed values for $|V|$.The values computed for all networks have been normalized between 0 and 1. The boxes span from the first to the third quartile of each data series, with an inner line representing the median. Lower and upper whiskers represent minimum and maximum observed values respectively.

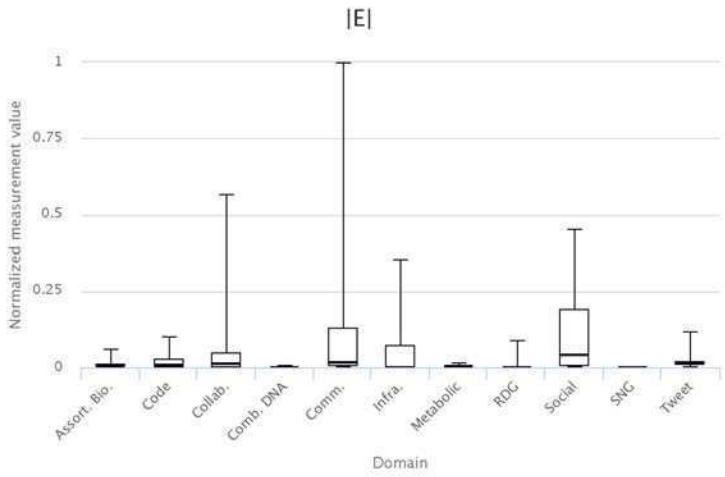


Figure 4.25: Statistical distribution of the observed values for $|E|$

Although not strictly the subject of this thesis, the figures allow for some interesting analysis. For example, it can be seen that social networks present a larger edge to vertex ratio than communication networks, or that metabolic networks seem to present a compact distribution, since neither maximum $|V|$ nor $|E|$ are far from the quartile box.

Figure 4.26 displays the values for the Maximum Degree. Communication, Social and Tweet networks present the most extreme values, and the Tweet networks are the ones with the higher first quartile, indicating that the Maximum Degree might be a candidate to help differentiate this domain from the rest.

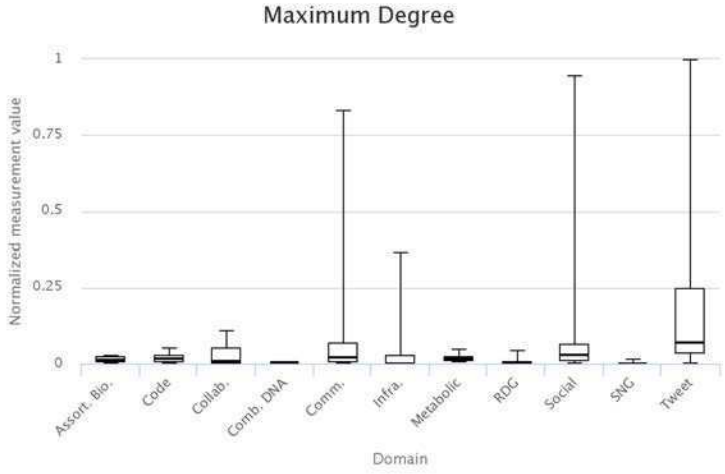


Figure 4.26: Statistical distribution of the observed values for Maximum Degree

Figure 4.27 displays the Diameter of the different networks considered. Although most of the domain "boxes" are in the same range in the chart, Metabolic networks stand out with a much higher first quartile, making this measure an interesting candidate to detect the domain.

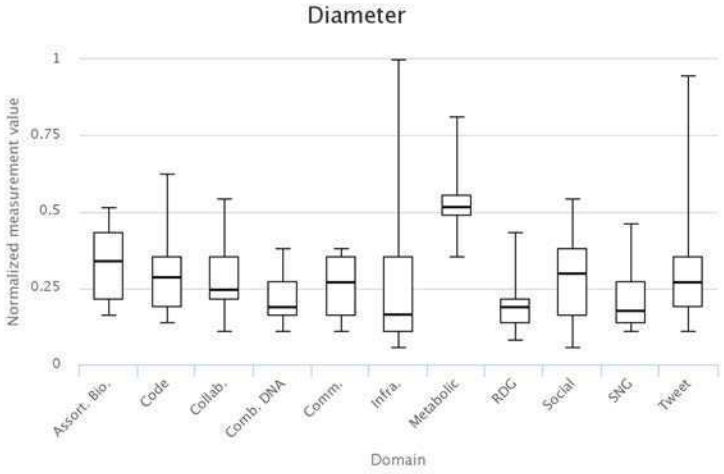


Figure 4.27: Statistical distribution of the observed values for Diameter

Figure 4.28 displays the values of the Path Length in each domain. Since most of them had been show to present a high correlation with the Diameter in section 4.3.1, it is no surprise to see once again the distribution along the same values, except for the Metabolic domain, which we can expect to be identifiable by means of this measure.

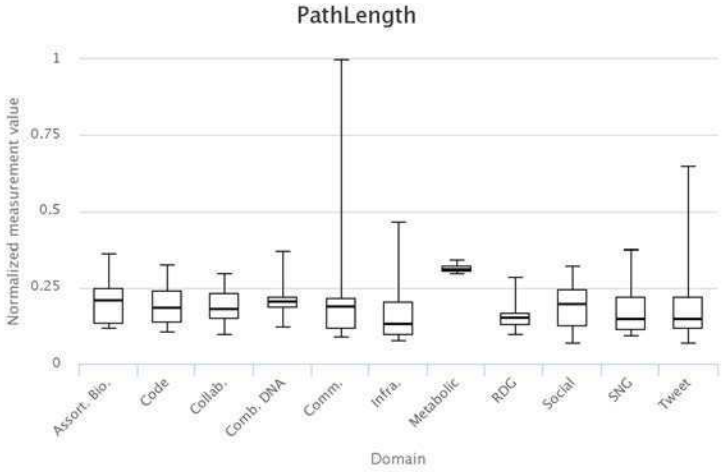


Figure 4.28: Statistical distribution of the observed values for PathLength

Figure 4.29 displays the values for the Codelength measure previously presented. Despite the fact that no single domain stands in a different section of the value range, we can see an interesting differentiation in the distribution of some domains. The quartile box for the Metabolic domain does not interset with any of those for Combinatorial DNA Library Synthesis, Reaction Dependency Graph nor Species Network Graph. The same can be said for the Social Domain.

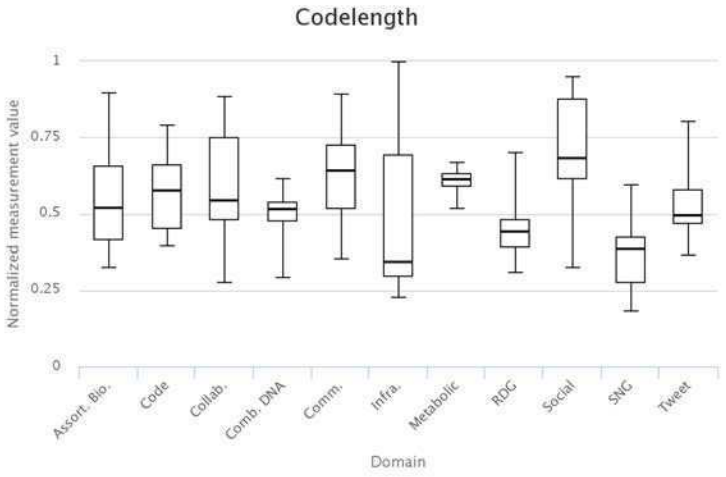


Figure 4.29: Statistical distribution of the observed values for Codelength

Figure 4.30 displays the values for the Offdiagonal Complexity. As in the previous chart, some clear distinctions appear, that might help an automatic system break ties when trying to identify different domains. Metabolic, Social and Tweet networks present a separate distribution from networks from the Combinatorial DNA Library Synthesis, Reaction Dependency Graph and Species Network Graph domains.

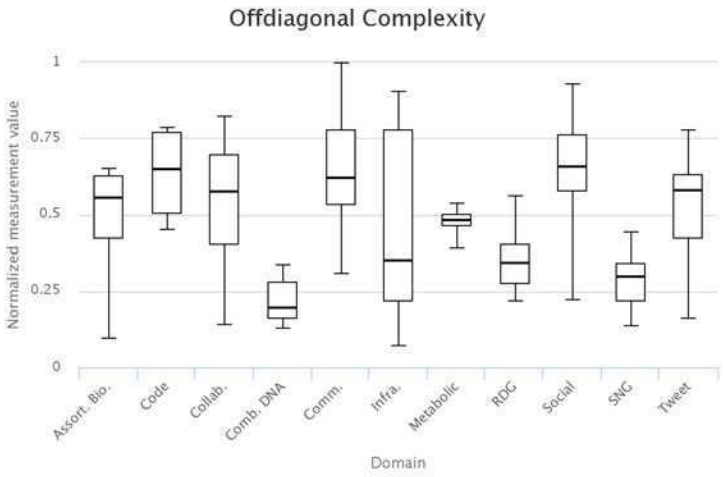


Figure 4.30: Statistical distribution of the observed values for Offdiagonal Complexity

Figure 4.31 displays the Global Efficiency computed for the different domains. Here, Tweet networks dominate the sprectrum of values observed, making the rest of the domains indistinguishable. The Combinatorial DNA Library Synthesis domain is the only other that shows a first quartile that rises above the base line of the chart, while Metabolic and Reaction Dependency Graph show no variation at all in their value range.

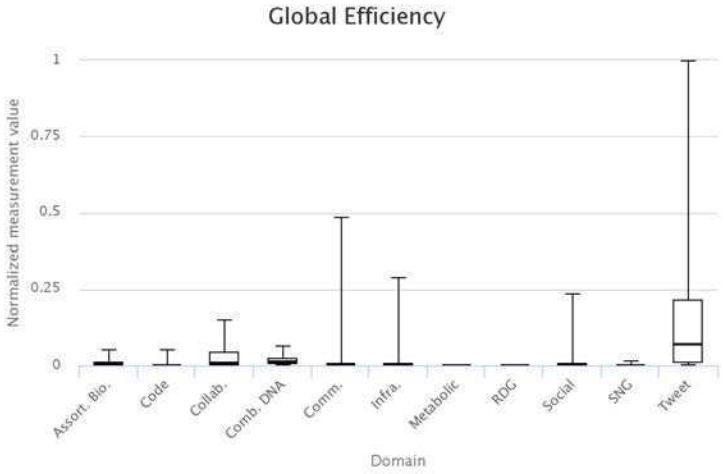


Figure 4.31: Statistical distribution of the observed values for Global Efficiency

We observe now the results of computing the measures for the different concepts that apply to individual vertices, which has allowed us to consider

a set of derived measures for each.

Betweenness Centrality will be reviewed first. As before, we present the normalized value distribution for the domains under study.

Figure 4.32 displays the values for the average Betweenness Centrality. Code and Social networks present a similar distribution, and the Metabolic networks display a distribution that slightly distinguishes it from the other domains.

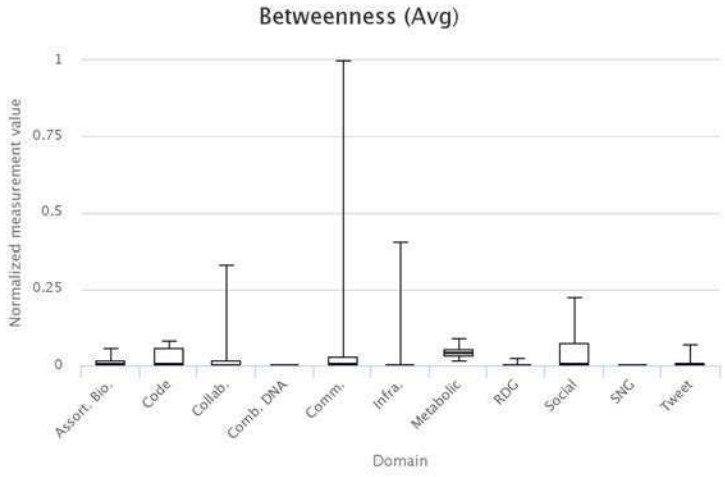


Figure 4.32: Statistical distribution of the observed values for average Betweenness Centrality

Figure 4.33 displays the values for the minimum Betweenness Centrality. As can be expected from a measure that identified minimum values, most of the domains present a flat distribution. Only Reaction Dependency Graph and Species Network Graph have a higher maximum value, but their complete quartile box remains on the base line of the chart. Figure 4.34 presents a similar scenario for the maximum Betweenness Centrality. Except for the Communication and Social networks, the domains present an equivalent value, which does not allow for discriminating between them.

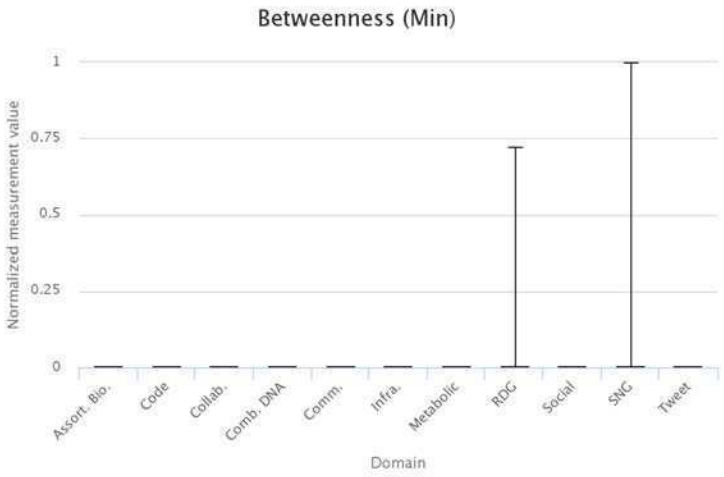


Figure 4.33: Statistical distribution of the observed values for minimum Betweenness Centrality

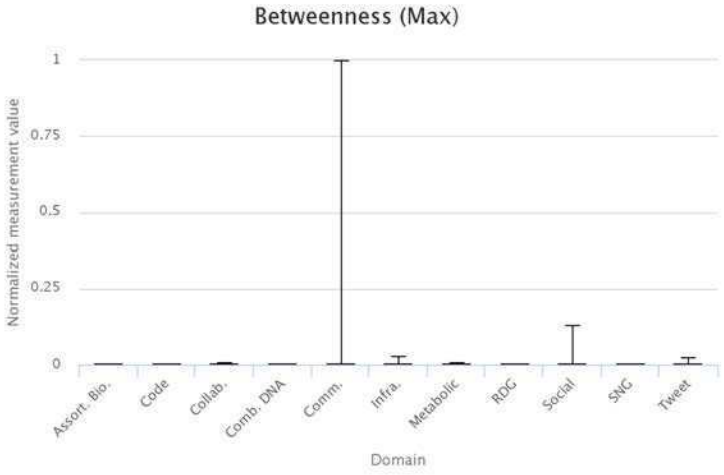


Figure 4.34: Statistical distribution of the observed values for maximum Betweenness Centrality

Figure 4.35 displays the values for the median Betweenness Centrality. Here the Metabolic domain presents a distinctive separation from the others.

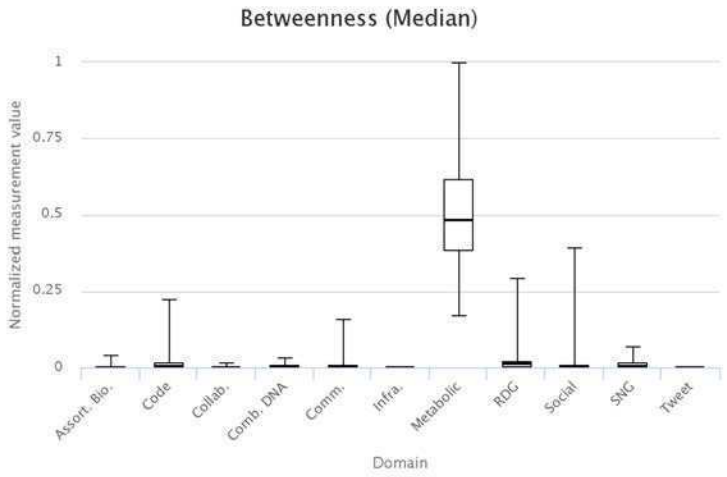


Figure 4.35: Statistical distribution of the observed values for median Betweenness Centrality

Figure 4.36, which contains the values for the first quartile Betweenness Centrality, allows for the same observation. In Figure 4.37 the third quartile Betweenness Centrality is shown. Here the Code networks present a more varied distribution, overlapping in values with the Metabolic domain, which means this specific complexity measure might not allow to distinguish between the two. Finally, Figure 4.38 displays the values for the inter-quartile range Betweenness Centrality, where Code and Metabolic domains overlap again, while the other domains present a compact distribution on the lower range of the observed values.

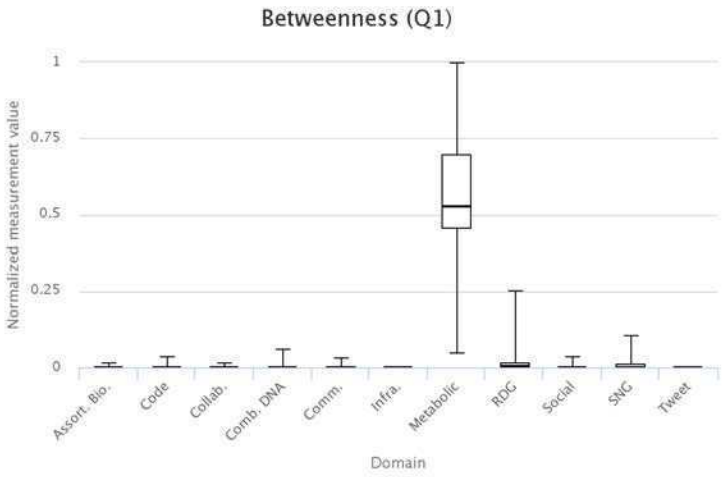


Figure 4.36: Statistical distribution of the observed values for first quartile Betweenness Centrality

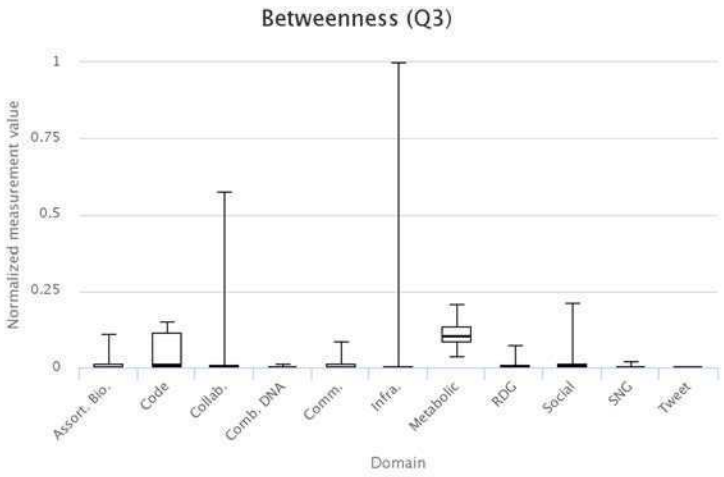


Figure 4.37: Statistical distribution of the observed values for third quartile Betweenness Centrality

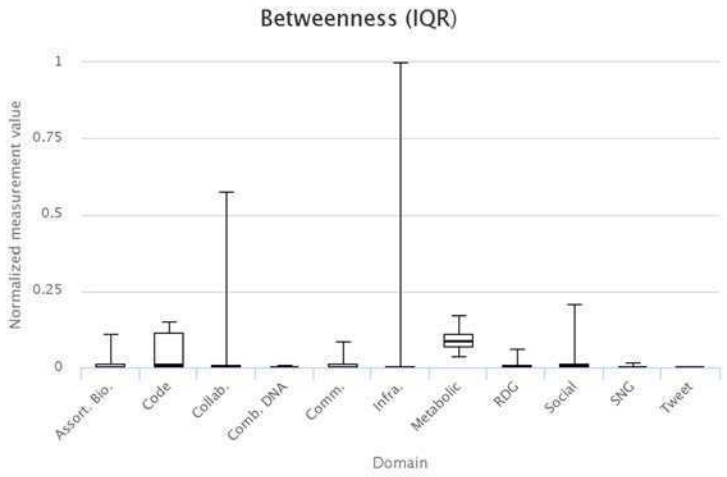


Figure 4.38: Statistical distribution of the observed values for inter-quartile range Betweenness Centrality

We focus now our attention on Figures 4.39 to 4.45, where the values for Closeness Centrality are presented.

In Figure 4.39 the distribution of the average values across all vertices in each network are presented. We can observe another clear distinction of the Metabolic networks, and this time Tweet networks are also clearly distinguishable, since their values are below almost all others. Although Combinatorial DNA Library Synthesis and Social networks present a more compact quartile box, they intersect with other domains, which might mean this measure will not be enough to set them apart.

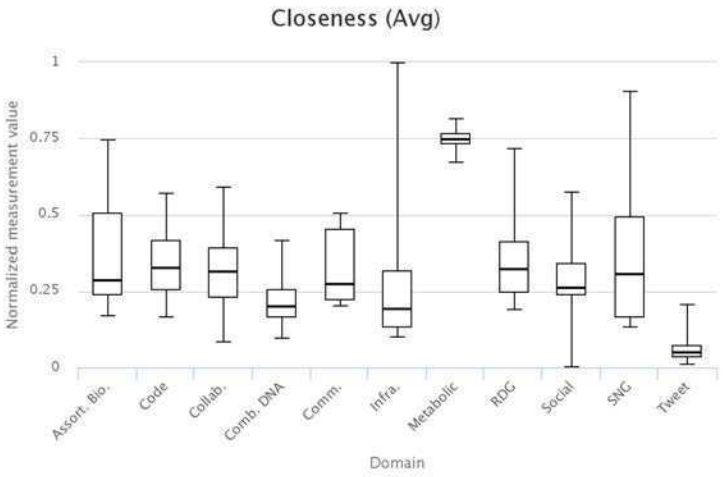


Figure 4.39: Statistical distribution of the observed values for average Closeness Centrality

Figure 4.40 displays minimum Closeness Centrality for the chosen domains. As with Betweenness Centrality, most categories present a low value range for this measure. The domain that stands out is Reaction Dependency Graph, with the both median an third quartile visible in the chart. Figure 4.41 displays a varied distribution for maximum Closeness Centrality, with only Metabolic networks standing out.

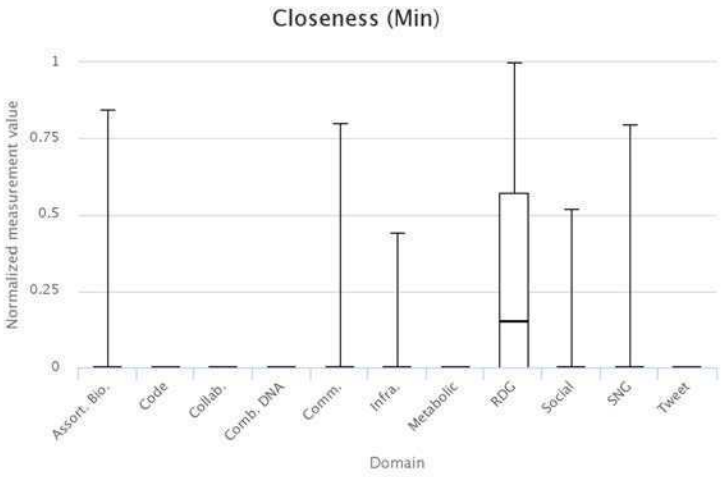


Figure 4.40: Statistical distribution of the observed values for minimum Closeness Centrality

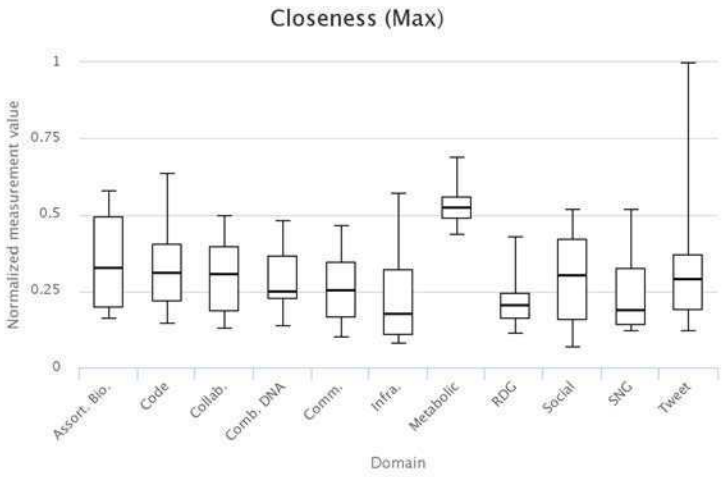


Figure 4.41: Statistical distribution of the observed values for maximum Closeness Centrality

Figure 4.42, which shows median Closeness Centrality, allows a clear distinction of the Metabolic domain. It also presents Tweet networks as a compact set of values in the lower value range, which allows it to be distingued from the other domains. Combinatorial DNA Library Synthesis and Code networks present a quartile box that does not overlap, allowing this measure to be a candidate to discriminate between the two.

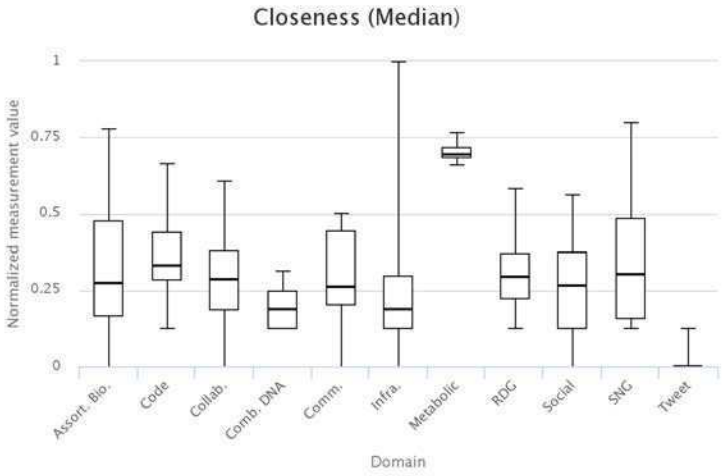


Figure 4.42: Statistical distribution of the observed values for median Closeness Centrality

In Figure 4.43 the first quartile Closeness Centrality has Metabolic net-

works once again represented in a distant segment of the values with respect to the other domains. Combinatorial DNA Library Synthesis networks also stand out, since they are almost out of the range observed for Code, Collaboration, Communication and Reaction Dependency Graph domains. Tweet networks are condensed on the bottom value of the chart.

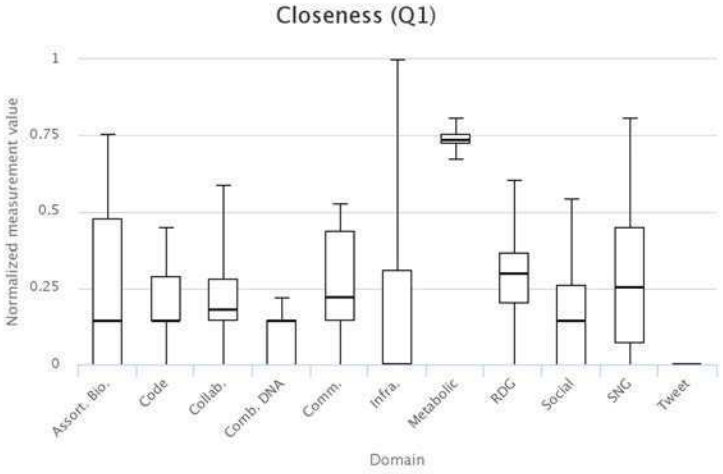


Figure 4.43: Statistical distribution of the observed values for first quartile Closeness Centrality

Figure 4.44 is for the third quartile Closeness Centrality. Metabolic and Tweet network are completely apart from each other, with not even an overlap in the extreme values.

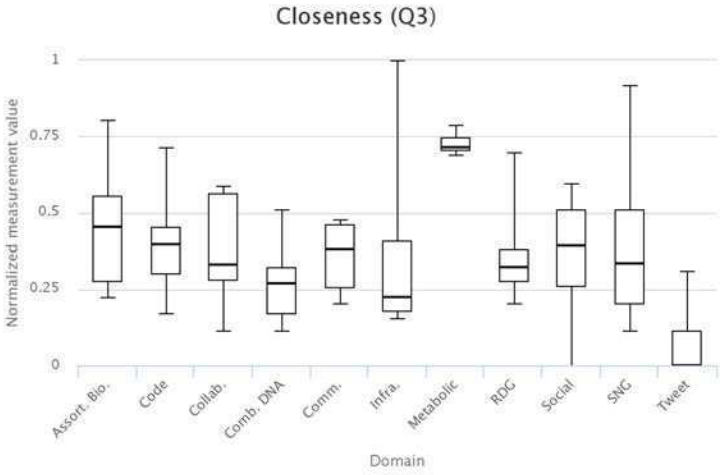


Figure 4.44: Statistical distribution of the observed values for third quartile Closeness Centrality

Finally, Figure 4.45 displays the values for the inter-quartile range Closeness Centrality. This measure is the least useful of those for this centrality, since there is overlap for almost all domains, with only Metabolic and Tweet networks presenting a slight distinction in their value ranges.

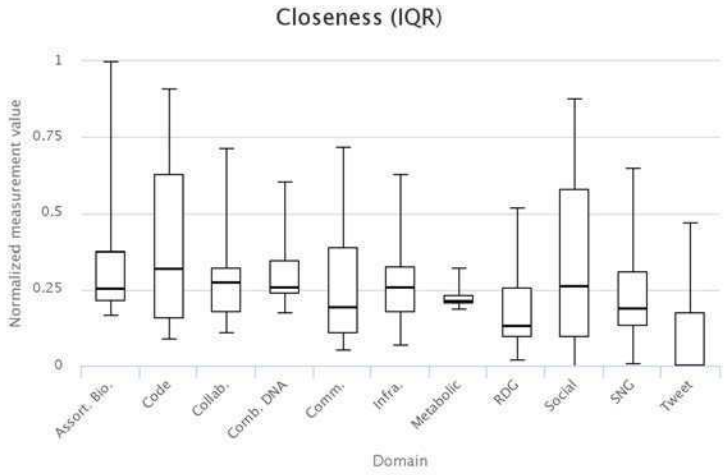


Figure 4.45: Statistical distribution of the observed values for inter-quartile range Closeness Centrality

To review the values obtained when measuring Clustering Coefficient, we display them in Figures 4.46 to 4.52.

We begin with Figure 4.46, where the average values for Clustering Coefficient allow a clear distinction of networks from the Reaction Dependency Graph domain. Combinatorial DNA Library Synthesis and Metabolic networks are apart from the other domains in their own flat distribution on the bottom values of the chart, with a similar distribution observed for Tweet networks.

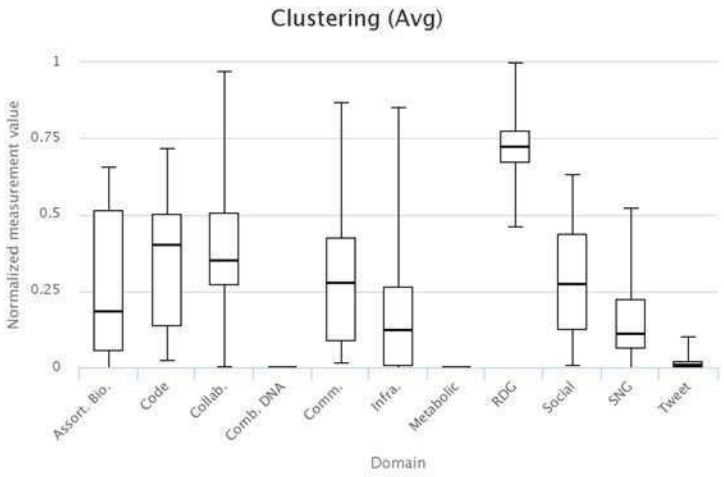


Figure 4.46: Statistical distribution of the observed values for average Clustering Coefficient

Figure 4.47 shows the minimum values observed for Clustering Coefficient, where Reaction Dependency Graph is the only domain to stand out in its general distribution.

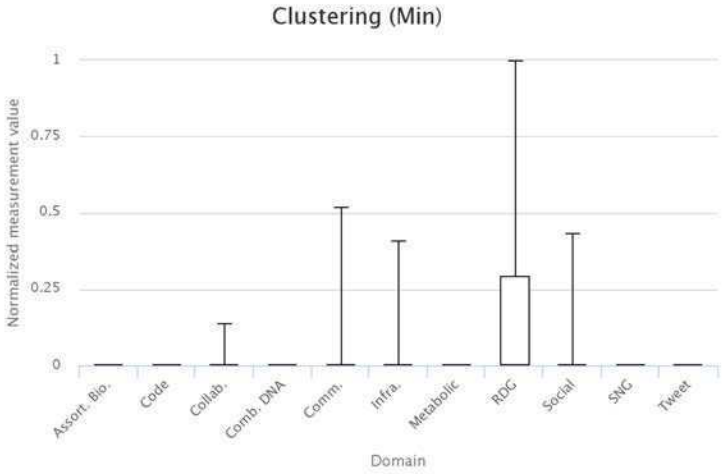


Figure 4.47: Statistical distribution of the observed values for minimum Clustering Coefficient

The maximum values presented in Figure 4.48 shows a completely different scenario. Combinatorial DNA Library Synthesis and Metabolic networks present only a small distribution with low values, Infrastructure and Species Network Graph networks both contain most values below the low

half of the observed values, while the other domains show predominantly to be distributed on the upper half of the values. The Reaction Dependency Graph domain seems to present most of its networks with the maximum value.

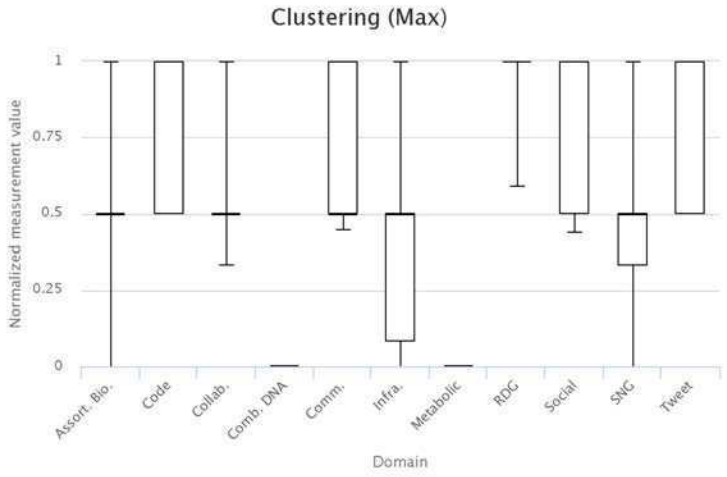


Figure 4.48: Statistical distribution of the observed values for maximum Clustering Coefficient

Figure 4.49, with the median values for Clustering Coefficient, displays a flat distribution of the Combinatorial DNA Library Synthesis, Metabolic and Tweet domains, a common value distribution for the other networks, with a lower overlap for Reaction Dependency Graph only in the lower values. A similar scenario is displayed in Figures 4.50 and 4.51, which show the first and third quartile values respectively. The inter-quartile range of the computed values for Clustering Coefficient, shown in Figure 4.52, keeps the flat distribution for Combinatorial DNA Library Synthesis, Metabolic and Tweet networks, but has a less clear distinction in the distribution of Reaction Dependency Graph.

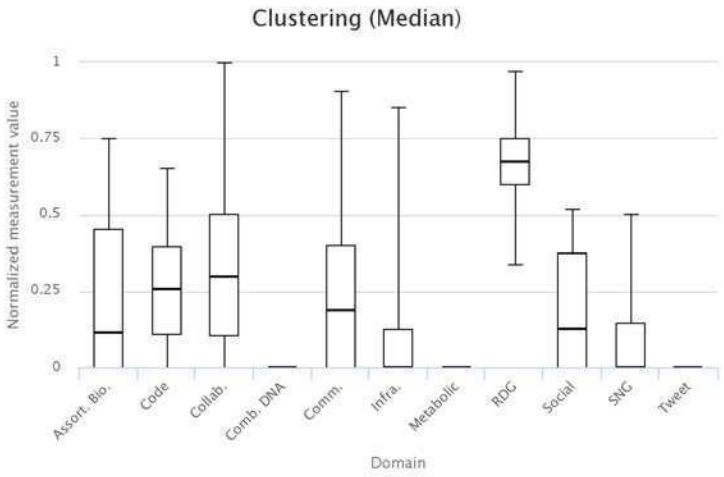


Figure 4.49: Statistical distribution of the observed values for median Clustering Coefficient

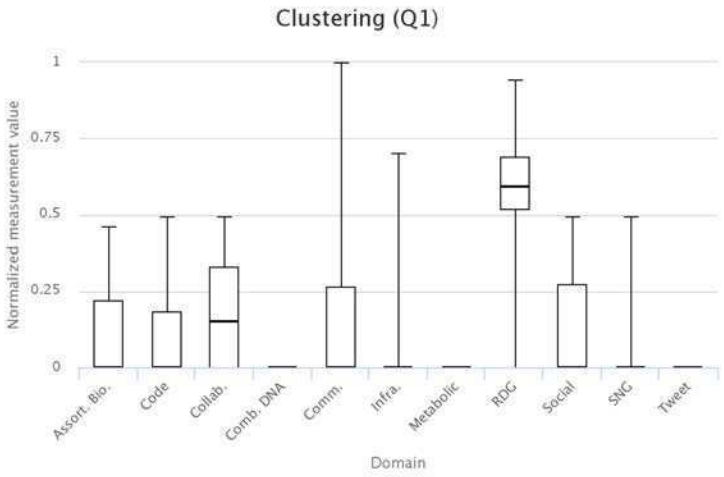


Figure 4.50: Statistical distribution of the observed values for first quartile Clustering Coefficient

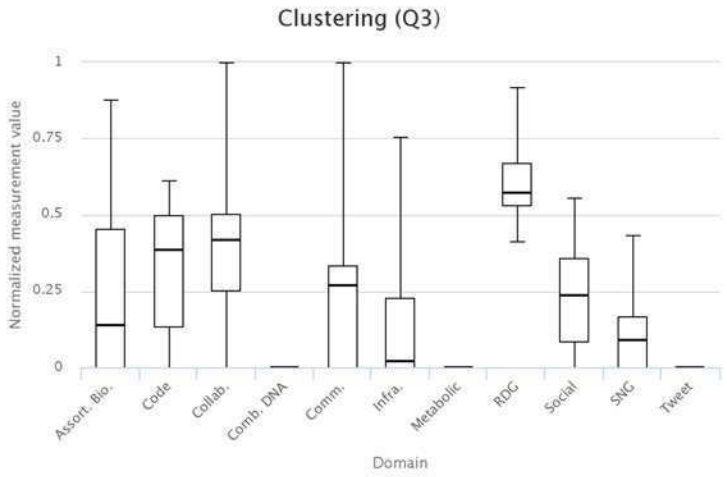


Figure 4.51: Statistical distribution of the observed values for third quartile Clustering Coefficient

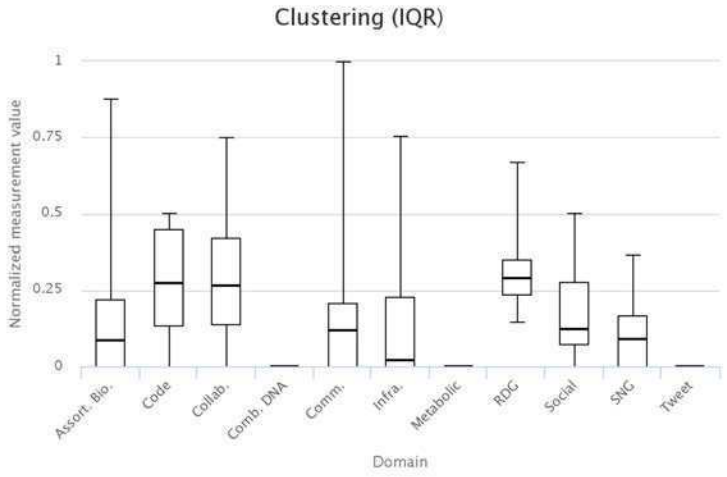


Figure 4.52: Statistical distribution of the observed values for inter-quartile range Clustering Coefficient

Figures 4.53 to 4.59 contain the distribution for the different measures related to Eigenvector Centrality.

The average values are presented in Figure 4.53. Reaction Dependency Graph and Species Network Graph domains show a similar distribution, with little overlap to the other domains. Combinatorial DNA Library Synthesis networks are observed to occupy the higher value range.

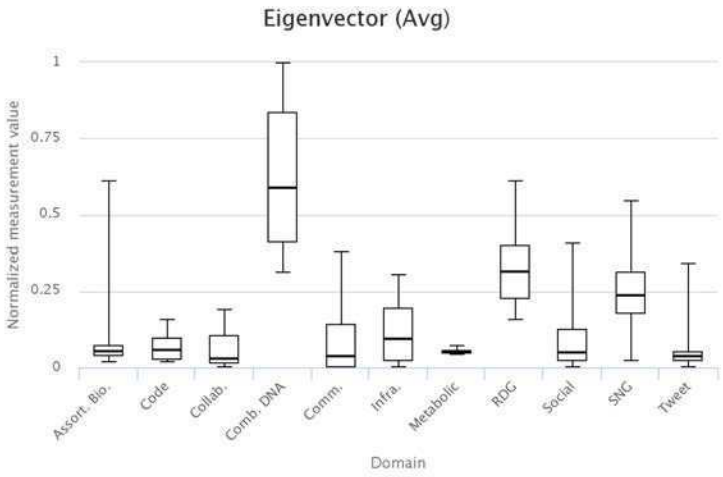


Figure 4.53: Statistical distribution of the observed values for average Eigenvector Centrality

The minimum values, displayed in Figure 4.54 show an almost complete distribution of the domains in the lower values of the chart, with only Reaction Dependency Graph reaching to slightly higher values. Figure 4.55 presents the maximum value distribution, where all networks have presented the maximum possible value. This is expected since the measure under study is in itself an average, which means that some vertex provides the maximum normalized Eigenvector Centrality, and thus is detected when considered for this chart.

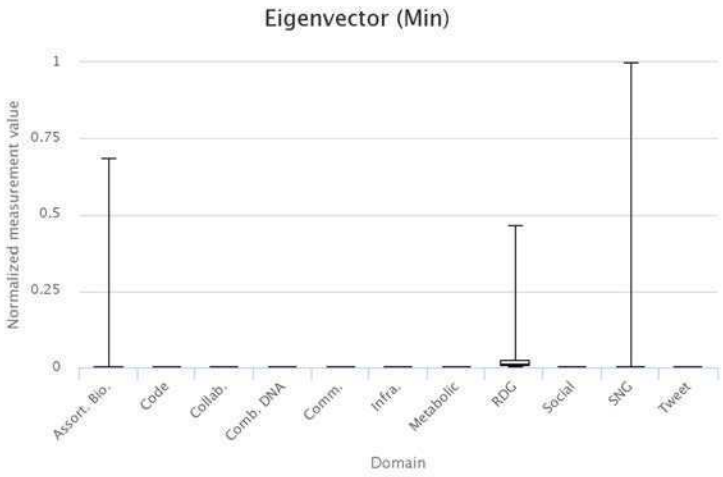


Figure 4.54: Statistical distribution of the observed values for minimum Eigenvector Centrality

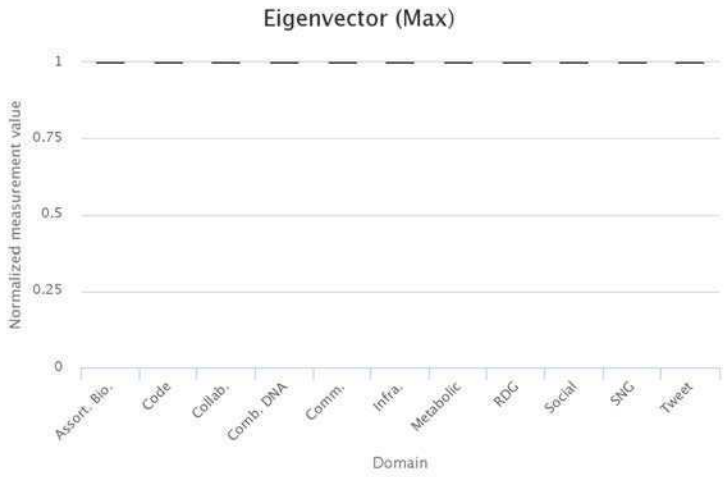


Figure 4.55: Statistical distribution of the observed values for maximum Eigenvector Centrality

The median values presented in Figure 4.56 are distributed similarly to the average values previously presented. Figure 4.57 shows that the first quartile values present a more compressed distribution of the same kind.

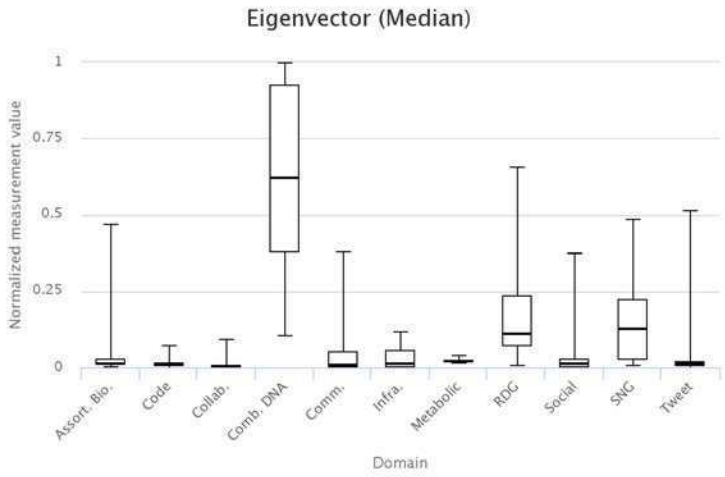


Figure 4.56: Statistical distribution of the observed values for median Eigenvector Centrality

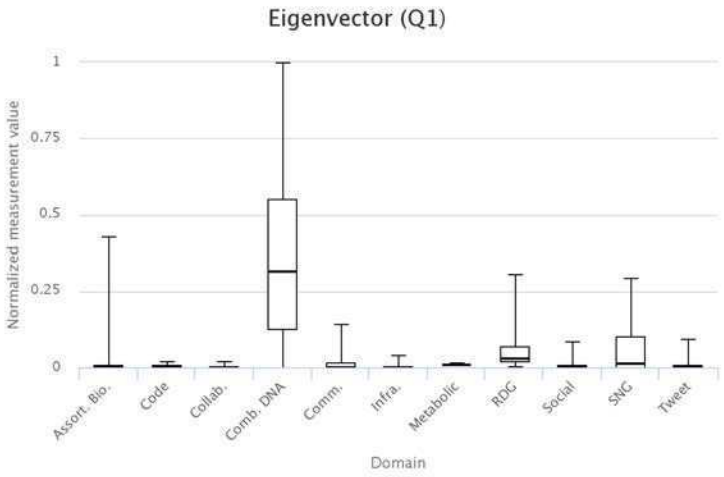


Figure 4.57: Statistical distribution of the observed values for first quartile Eigenvector Centrality

In Figure 4.58, the third quartile values for Eigenvector Centrality can be seen to keep a distinctive value range for Combinatorial DNA Library Synthesis networks, but in this chart networks in the Reaction Dependency Graph overlap with them. A similar analysis can be observed for the interquartile range, shown in Figure 4.59.

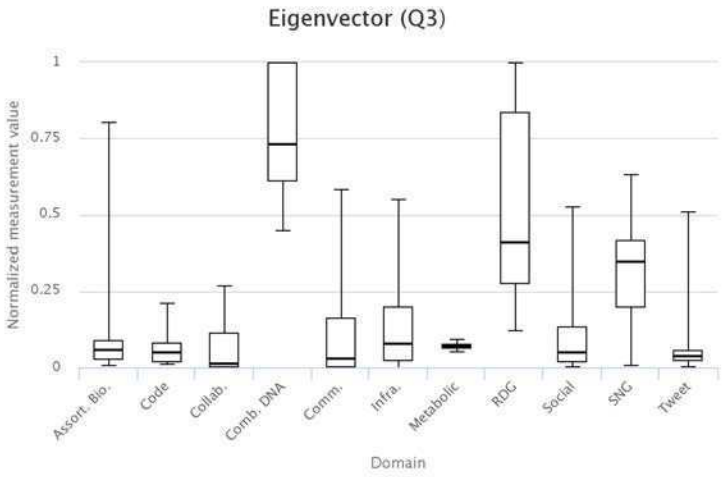


Figure 4.58: Statistical distribution of the observed values for third quartile Eigenvector Centrality

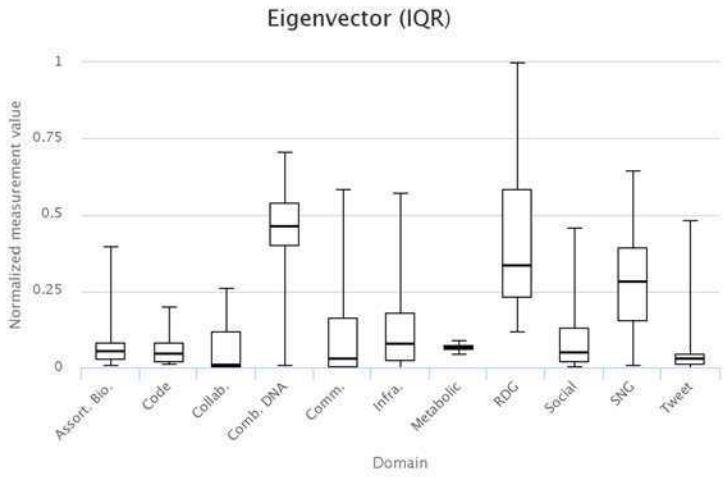


Figure 4.59: Statistical distribution of the observed values for inter-quartile range Eigenvector Centrality

The final set of complexity measures that have been computed for all vertices in all networks is PageRank. Figures 4.60 to 4.66 contain the different charts that study the values observed.

Figure 4.60 presents the value of the averaged PageRank values. There is a strong overlap in the values for all domains. This is mainly due to the fact that Infrastructure networks present a wider range of values, covering that observed for the others. The minimum and maximum PageRank distributions, shown in Figures 4.61 and 4.62 respectively, have an equivalent distribution for the different domains.

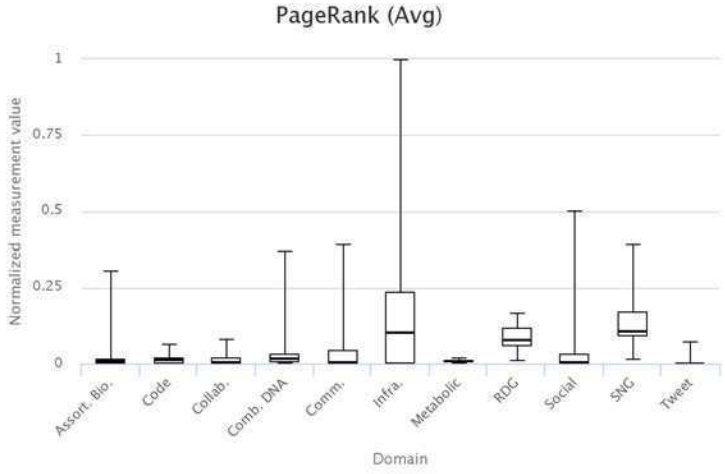


Figure 4.60: Statistical distribution of the observed values for average PageRank

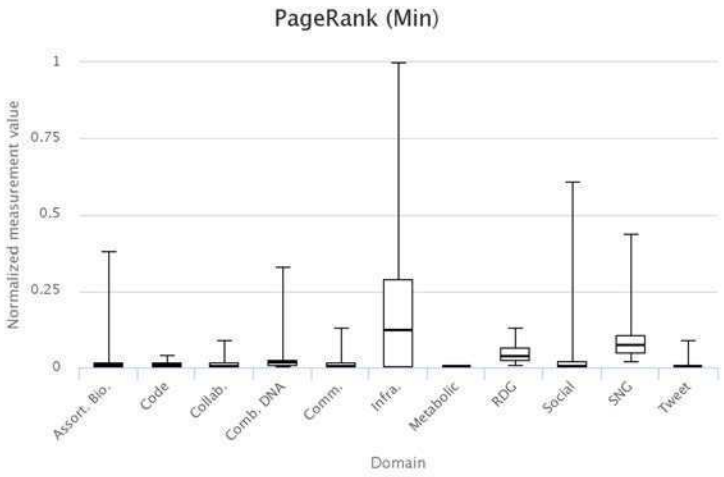


Figure 4.61: Statistical distribution of the observed values for minimum PageRank

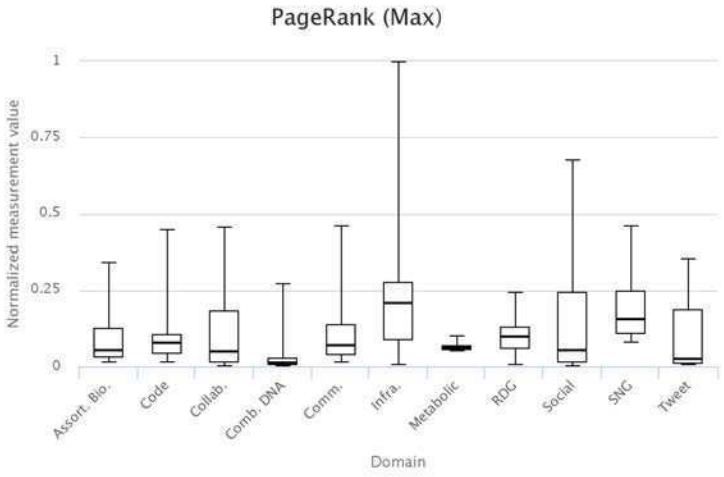


Figure 4.62: Statistical distribution of the observed values for maximum PageRank

Figures 4.63, 4.64, 4.65 and 4.66 repeat the pattern mentioned for the previous charts. They each display the values for median, first quartile, third quartile and inter-quartile range for the computed PageRank of the vertices.

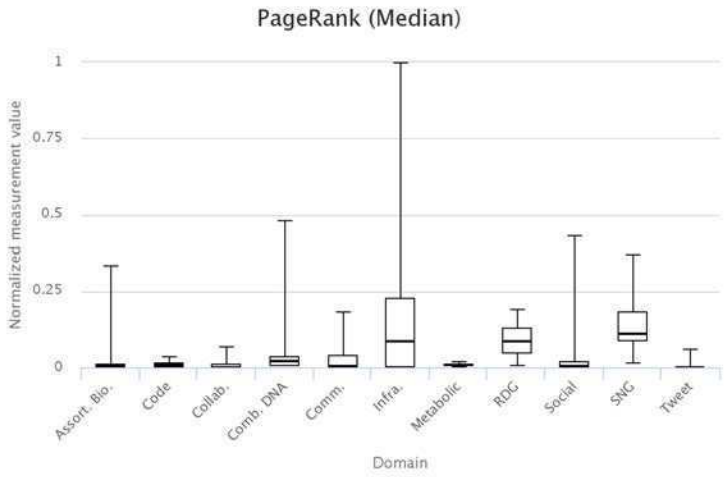


Figure 4.63: Statistical distribution of the observed values for median PageRank

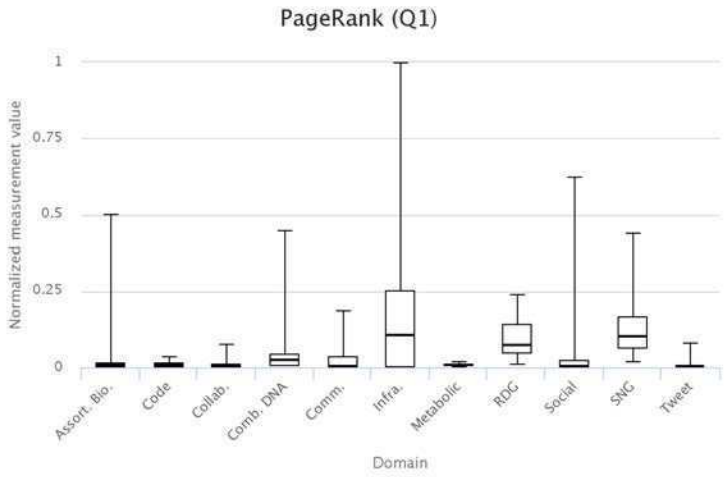


Figure 4.64: Statistical distribution of the observed values for first quartile PageRank

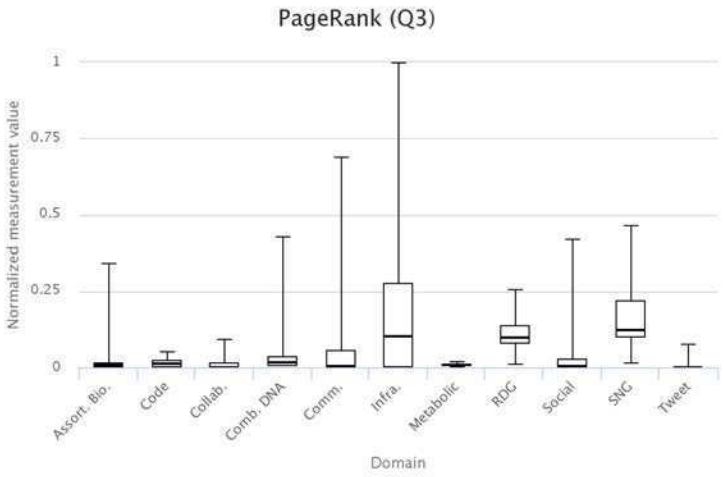


Figure 4.65: Statistical distribution of the observed values for third quartile PageRank

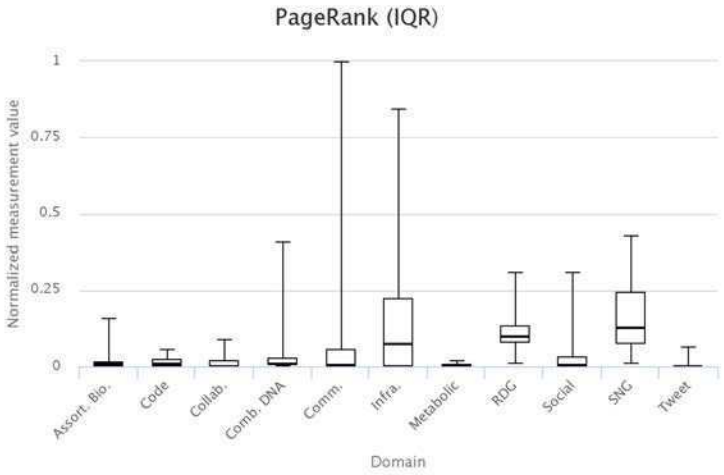


Figure 4.66: Statistical distribution of the observed values for inter-quartile range PageRank

From all of the distributions observed for the PageRank measure, the most distinctive feature is the fact that Species Network Graph and Reaction Dependency Graph domains, while indistinguishable between themselves, present little to no overlap to the other domains, except for Infrastructure.

4.4 Review

This chapter began with a list of 11 proposed domains, for studying a set of networks which were used by researchers in each of them, giving us 240 in total to analyze. Some of these domains share some degree of similarity in their field of study, like Reaction Dependency Graphs and Species Network Graphs, while others represent a completely separate concept, like Tweet networks.

After presenting each of them, and providing some visualization examples, a selection of 13 measures from those presented in Chapter 2 was made. Benefiting from the fact that some of them are vertex measures, we derived additional graph measures for their average, maximum, minimum, and quartile distributions, reaching in total 43 measures for each network.

These values, which were found as easy to compute, but with low overall correlation between them, will be used in following chapters to build experiments that apply classification algorithms, with the final objective of helping us determine if network structure can be used to identify the domain to which each belongs.

Although a variety of networks properties and features have been reported for a number of domains in the literature, to the best of our knowledge, no previous attempt has been made up to now to look in detail at the correlation between several of these measures within a domain data set, and across data sets from different domains. Thus the results presented in this chapter represent the addition of new knowledge, and could be used in the future to guide the first steps of even larger data-driven network automated classification.

Capítulo 4: Dominios de redes y métricas de complejidad

Este capítulo tiene dos objetivos. El primero es obtener suficientes redes de distintos dominios, preferentemente de situaciones del mundo real, y aplicar una clasificación manual que las separe de acuerdo a su origen. En capítulos siguientes intentaremos usar clasificación automática para determinar si las redes de un mismo dominio comparten valores similares para alguna de las métricas de complejidad analizadas en el capítulo 2.

El segundo objetivo es seleccionar un subconjunto de las métricas presentadas en la sección 2.4. Estos valores son calculados para todas las redes elegidas, y se presenta información estadística para cada caso. Observando la distribución de los valores analizamos la factibilidad de usarlos para discriminar entre los distintos dominios estudiados. En los próximos capítulos estos valores calculados serán el valor de entrada de los algoritmos de clasificación, los cuales intentarán distinguir automáticamente el dominio al que pertenece cada red.

En la sección 4.2 se presentan los 11 dominios elegidos:

- Biológicas varias
- Código
- Colaboraciones
- Síntesis automática de bibliotecas de ADN
- Comunicación
- Infraestructura
- Metabólicas
- Grafos de dependencia de reacciones
- Sociales
- Grafos de redes de especies
- Tweets

En total se proponen 240 redes para analizar. Algunos dominios presentan similitudes en su campo de estudio, como los grafos de dependencia de reacciones (RDG) y los grafos de redes de especies (SNG). Por otro lado, otros dominios son completamente independientes, como el caso de las redes en base a Tweets.

La sección 4.3 presenta las 13 métricas elegidas, y aprovechando que algunas de ellas aplican a cada vértice en un grafo, se derivan métricas adicionales en base al promedio, mínimo, máximo, y cuartiles de las mismas. En total disponemos de 43 métrias para cada grafo.

Se presenta un análisis de la correlación de los valores obtenidos por cada dominio, así como también considerando el total de grafos.

Luego realizamos una comparación de los tiempos de procesamiento teóricos de las métricas con el tiempo relativo total que fue necesario para calcularlas.

Finalmente, presentamos gráficos "boxplot" donde para cada métrica se compara la distribución estadística de los valores obtenidos, agrupados por dominio. Observando los solapamientos en estas imágenes es posible vaticinar la posibilidad (o imposibilidad) de que un algoritmo de clasificación pueda diferenciar los dominios de grafos en base a las métricas calculadas.

Chapter 5

Validating Classification Ability

5.1 Description

In this chapter we propose a custom classification of automatically generated networks. They are to be used to validate the capacity of the proposed algorithms, regarding their ability to offer an insightful partitioning of the data provided to them.

First we will detail the source of a technique that allows creating arbitrary artificial instances of networks, where some of their attributes can be specified.

Then we will review the statistics for the generated networks, and finally they will be analyzed using the algorithms proposed in Chapter 3. The results of these classification experiments will set the expectations for equivalent processes, to be applied in the next chapter, to the real network reviewed in Section 4.2.

5.2 Creating Artificial Networks

5.2.1 CiGRAM

In order to ensure that the ideas mentioned so far as regards network measurements and classification algorithms are sound, we need to test them in a "controlled" environment. This implies that before applying any process to the networks researched and presented in Section 4.2, the algorithms should be tested against networks where the measures and classes are well organized and known beforehand, without any room for misclassification in the original sources, nor misinterpretation of vertex or edges data. It would

also be beneficial that this initial benchmark set of networks be based on a scheme that allows the generation of as many networks as desired, preferably allowing any number of networks with similar characteristics to be built, so that we are allowed to test different aspects of the ideas proposed.

Fortunately, there is such a way. We define now CiGRAM, as presented at http://cigram.ico2s.org/.

Definition 20. ***CiGRAM (Circular Gaussian Random grAph Model)*** *is a generator for complex networks with configurable topology. It is able to model flexible node degree distributions, degree assortativity and community structure. Its overreaching objective is to create graphs topologically similar to real-world networks.*

With this utility, random networks can be built, always ensuring that some specific properties are obtained.

- Number of nodes
- Graph density $[0, 1]$
- Number of communities
- Assortativity $[-5, 5]$

To give an idea of the different structures that can be obtained using CiGRAM, Figure 5.1 shows an assortment of networks with different sizes built with the application. In Figure 5.2 the variation when density changes can be observed. As the number of communities changes, instances like those presented in Figure 5.3 are built. The effect of assortativity is shown in Figure 5.4.

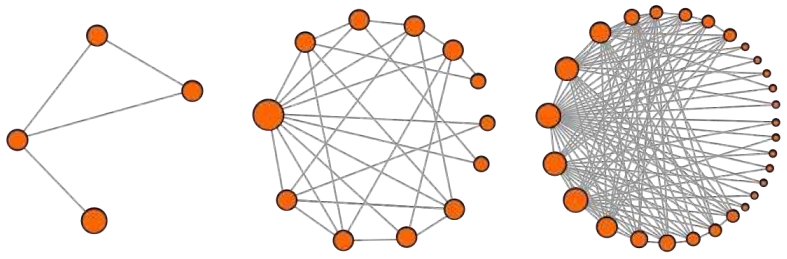


Figure 5.1: Examples of CiGRAM networks with different number of nodes

Also, for more advanced experiments, the following can also be defined:

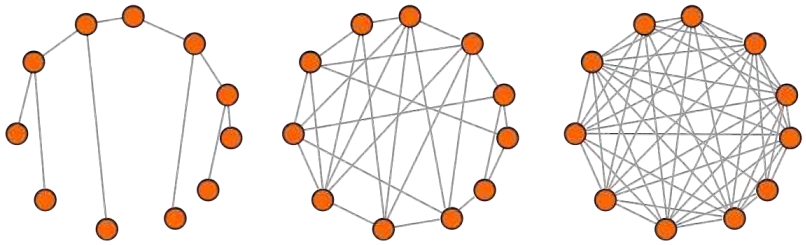


Figure 5.2: Examples of CiGRAM networks with different density

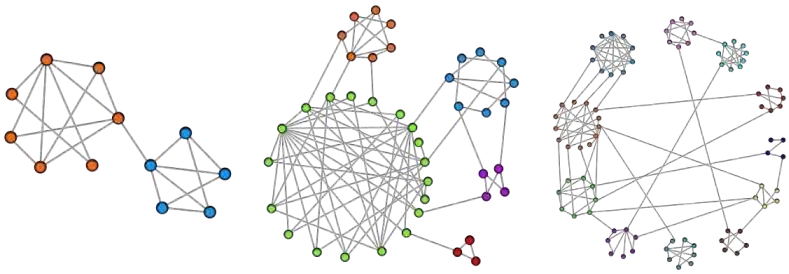


Figure 5.3: Examples of CiGRAM networks with different number of communities

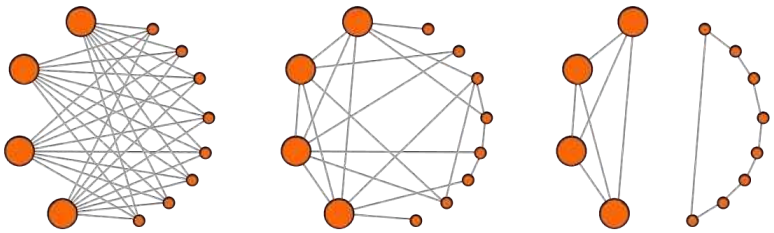


Figure 5.4: Examples of CiGRAM networks with different assortativity

- Standard deviation of the node positions
- Standard deviation of the scores of each node position
- Fraction of edges between communities $[0, 1]$
- Probability of community overlapping $[0, 1]$
- Standard deviation of the community positions
- Standard deviation of the scores of each community position

Apart from using the online visual tool to create networks, the source code is available, which allows for custom tools to be built upon this generator.

5.2.2 Generated Networks in Custom Classes

In order to benefit from the features provided by CiGRAM, while keeping a manageable number of classes and instances to analyze, we propose the creation of the following networks, detailed in Table 5.1.

Table 5.1: Artificial Networks to be created with CiGRAM

Attribute	Possible Values
Number of Vertices	100, 1000
Graph Density	0.1, 0.9
Number of Communities	10, 50
Random Seed	1...20

From the values in the table, it can be seen that 3 attributes will vary between 2 possible values, giving raise to 8 different classes, where each is represented by 20 different instances. With this simple configurations, we have now access to a set of 160 networks that present a similar distribution to the researched ones presented in Section 4.2. This is crucial to our experiments, since by considering an equivalent number of "domains", each with a comparable number of networks as those presented in previous chapters for the real networks, will allow us to validate the capabilities of the classification algorithms to work in such an environment and produce useful results. With these artificial networks in hand, we can proceed to testing our ideas using the algorithms detailed in Section 3.3.

5.3 Generated Networks Statistics

Figure 5.5 presents the correlation between the selected measures for all of the networks generated with the ranges defined in the previous section.

As can be seen, there is a higher level of dependency between these values, owing to the fact that they were created with a strict set of parameters. This fact must be considered when analyzing any differences in behavior between the artificial networks vs that of the researched ones.

Figures 5.6 to 5.15 allow us to review the distribution among the proposed artificial classes for some of the chosen measures. The domains presented in the charts correspond to the 8 parameters variations proposed.

	|V|	|E|	CL	Diam	PL	Betw	Close	GE	MD	OdC	Clust	Eigen	PR
|V|	1,00	0,66	0,87	0,34	0,20	0,63	0,42	-0,34	0,83	0,95	0,04	-0,12	-1,00
|E|	0,66	1,00	0,81	-0,37	-0,47	-0,16	-0,35	-0,47	0,95	0,60	0,41	0,46	-0,66
CL	0,87	0,81	1,00	-0,06	-0,22	0,31	0,05	-0,72	0,91	0,86	0,21	0,32	-0,87
Diam	0,34	-0,37	-0,06	1,00	0,94	0,83	0,93	0,43	-0,17	0,36	-0,69	-0,88	-0,34
PL	0,20	-0,47	-0,22	0,94	1,00	0,75	0,93	0,53	-0,29	0,23	-0,70	-0,92	-0,19
Betw	0,63	-0,16	0,31	0,83	0,75	1,00	0,91	0,03	0,12	0,63	-0,38	-0,64	-0,63
Close	0,42	-0,35	0,05	0,93	0,93	0,91	1,00	0,25	-0,11	0,45	-0,60	-0,81	-0,42
GE	-0,34	-0,47	-0,72	0,43	0,53	0,03	0,25	1,00	-0,51	-0,38	-0,41	-0,71	0,35
MD	0,83	0,95	0,91	-0,17	-0,29	0,12	-0,11	-0,51	1,00	0,79	0,33	0,31	-0,83
OdC	0,95	0,60	0,86	0,36	0,23	0,63	0,45	-0,38	0,79	1,00	-0,09	-0,15	-0,95
Clust	0,04	0,41	0,21	-0,69	-0,70	-0,38	-0,60	-0,41	0,33	-0,09	1,00	0,69	-0,05
Eigen	-0,12	0,46	0,32	-0,88	-0,92	-0,64	-0,81	-0,71	0,31	-0,15	0,69	1,00	0,12
PR	-1,00	-0,66	-0,87	-0,34	-0,19	-0,63	-0,42	0,35	-0,83	-0,95	-0,05	0,12	1,00

Figure 5.5: Correlation between the selected measures for the generated networks

They are named according to the following schema:

`n[Number of Vertices]-d[Graph Density]-k[Number of Communities]`

Figure 5.6 shows the values for Diameter. Here we can begin to observe how a set of artificially built networks can create a distinct set of measures. The values for the classes with density 0.9 all appear around the lower quarter of the chart, with the classes that have density 0.1 appear over them, with even some distance between each. A classification algorithm that could take advantage of these value distribution could clearly distinguish at least 2 artificial classes, based on their density.

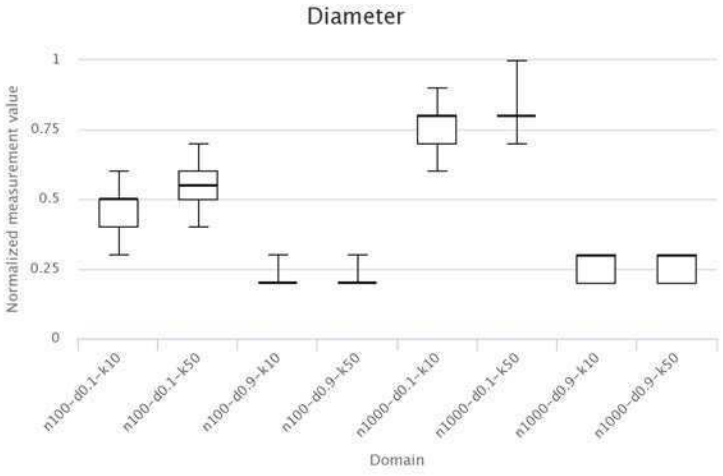


Figure 5.6: Statistical distribution of the observed values for Diameter in the artificial networks generated

Figure 5.7 shows the values for PathLength. Because the density was one of the parameters applied to generate the different classes, once again

we can observe a distinct distribution of the values in thie measure. We can also appreciate that there is almost no overlap in values between the classes that have 100 and 1000 vertices.

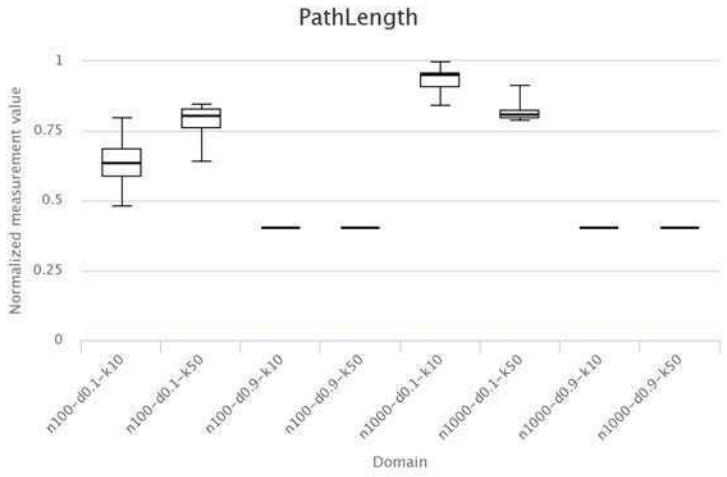


Figure 5.7: Statistical distribution of the observed values for PathLength in the artificial networks generated

Figure 5.8 present the distribution of the Codelength measure for the artificial networks. These results will surely be a boon for our experiments, since there is a clear step-increase for each class. We can propose now that Codelength allows to distinguish artificial networks of different number of vertices and densities, although it does not easily differentiate them by number of communities.

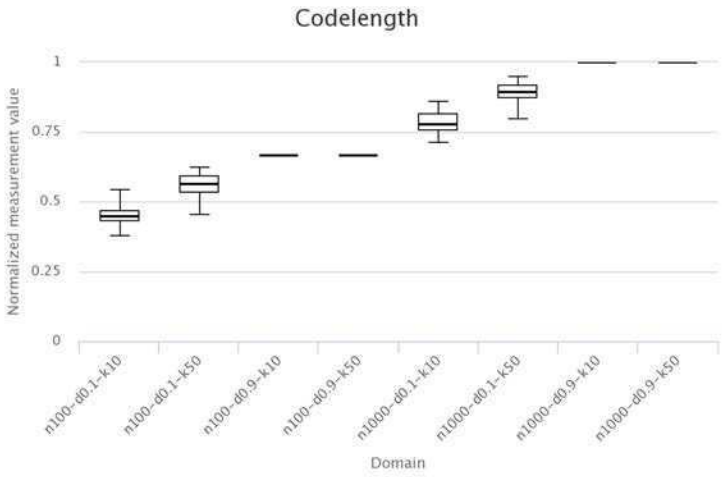


Figure 5.8: Statistical distribution of the observed values for Codelength in the artificial networks generated

Figure 5.9 presents the values for Offdiagonal Complexity. Although there is overlap for all classes with the same number of vertices, this measures provides a clear way of distinguishing at least 2 groupings of classes: those with 100 and those with 1000 vertices. It is a great example of a measure that provides a solution to part of the classification problem, suggesting that a combination of complexity measures with low correlation might provide a good classification ability.

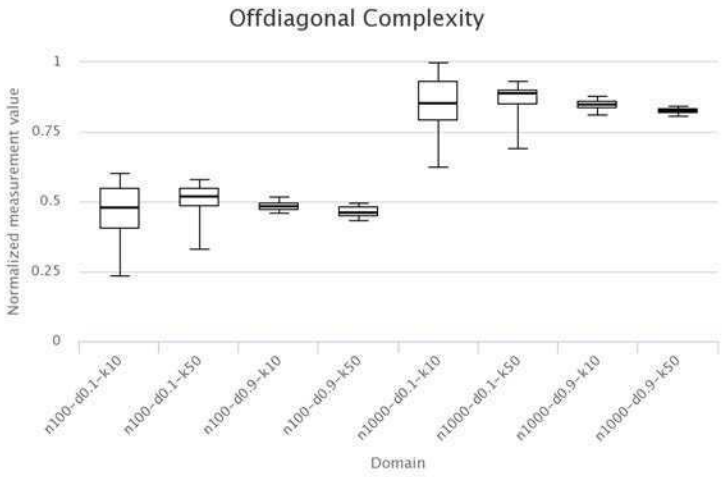


Figure 5.9: Statistical distribution of the observed values for Offdiagonal Complexity in the artificial networks generated

The Global Efficiency of the artificial networks, presented in Figure 5.10, shows a clear distinction between the classes, but only when their density is low. We can observe that there is almost no overlap in the quartile boxes for the classes with density 0.1, but all networks with density .9 are given the same value. This raises the question of the range of values in the artificial generation parameters that might create networks that are clearly distinguishable, and those that might confuse a classification algorithm.

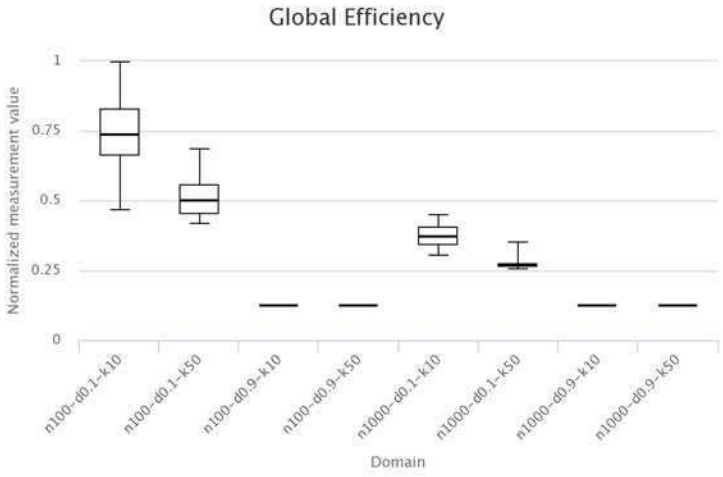


Figure 5.10: Statistical distribution of the observed values for Global Efficiency in the artificial networks generated

Figure 5.11 shows the values for Betweenness Centrality of the networks, considering the average value over all vertices. We can observe than only the classes with density 0.1 and 1000 vertices are away from the general distribution.

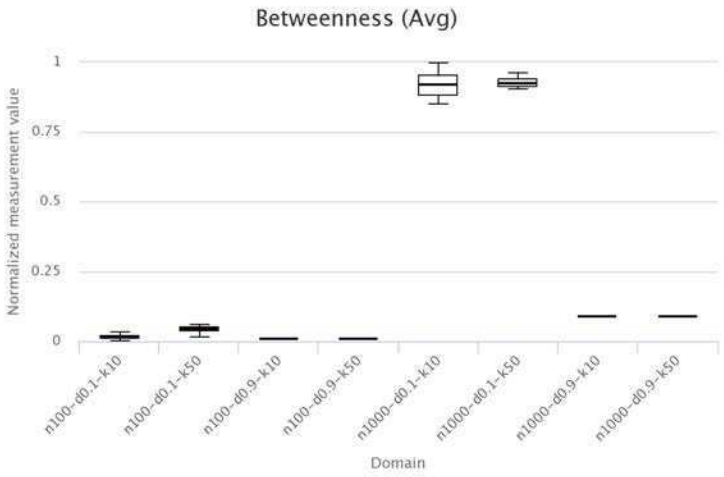


Figure 5.11: Statistical distribution of the observed values for average Betweenness Centrality in the artificial networks generated

The distribution averaged Closeness Centrality is shown in Figure 5.12, with a striking resemblance to PathLength. This is not surprising since the correlation analysis have already pointed out the similarities between both.

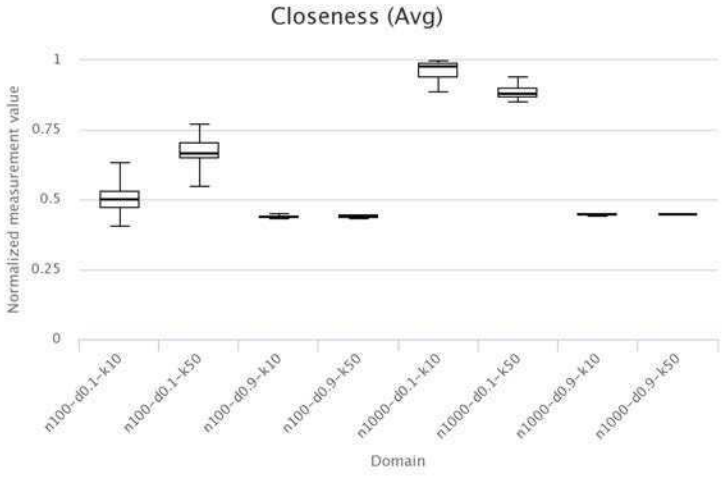


Figure 5.12: Statistical distribution of the observed values for average Closeness Centrality in the artificial networks generated

Figure 5.13 shows the values for Clustering Coefficient, averaged over all vertices for each network. There is an almost perfect match in the quartiles boxes for the classes that share number of vertices and diameter. This allow to conjecture that the number of communities in a network does not affect

the Clustering Coefficient, or at least not its average.

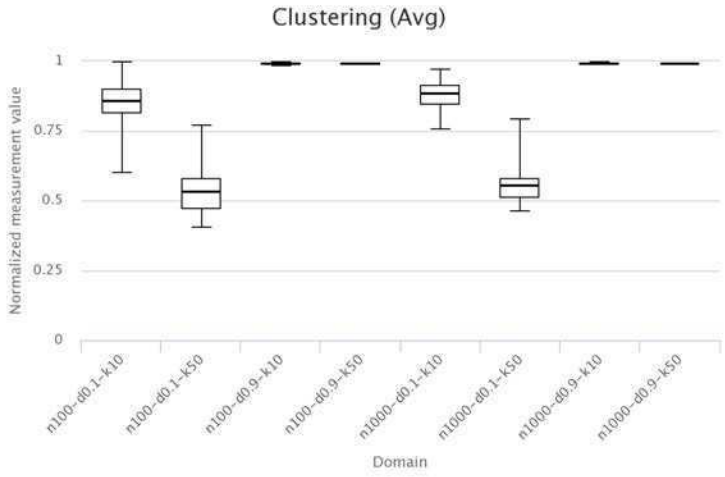


Figure 5.13: Statistical distribution of the observed values for average Clustering Coefficient in the artificial networks generated

Average Eigenvector Centrality, which had also been shown to present a high correlation (although negative) to PathLength, is presented in Figure 5.14. This measure seems to allow only to distinguish the density of the different classes.

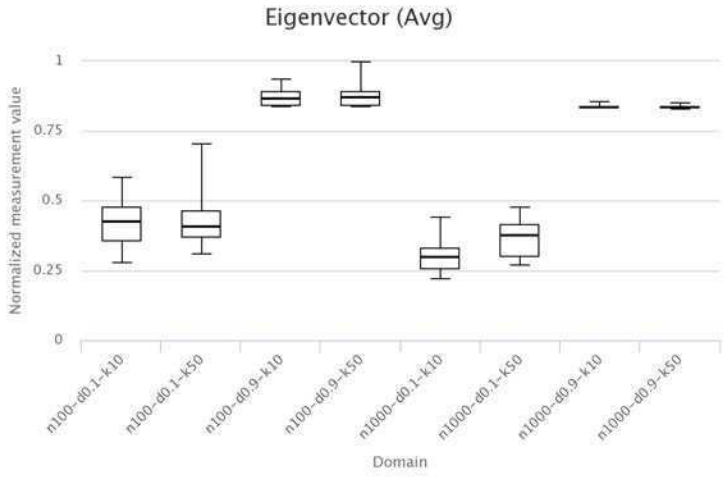


Figure 5.14: Statistical distribution of the observed values for average Eigenvector Centrality in the artificial networks generated

Finally, we present in Figure 5.15 the values for average PageRank. From the correlation analysis we could already predict some of the distribution

observed here, since it has a strong negative correlation to the number of vertices and also to the Offdiagonal Complexity.

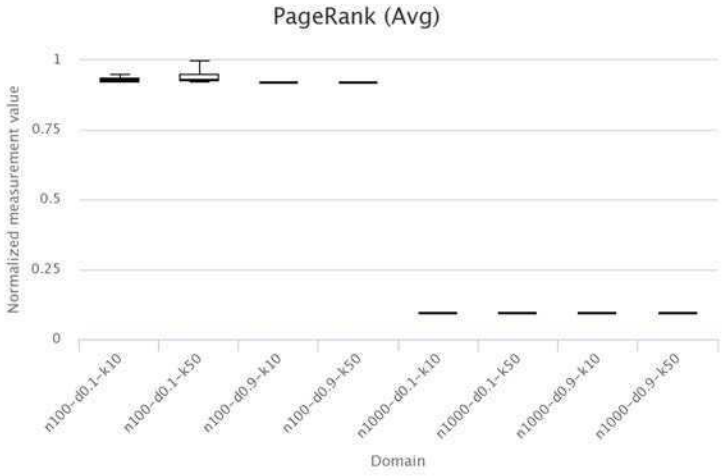


Figure 5.15: Statistical distribution of the observed values for average PageRank in the artificial networks generated

The parameters that guide the construction of these networks seem to impose a structure that is reflected in a greater rigidity for most of the values presented. This fact, far from being a complication, allows a clearer view of the possibility of classifying networks via the measurement of the complexity measures proposed. If an algorithm such as those proposed in chapter 3 is to discriminate between families of instances with attributes in defined ranges, then the first step must be to confirm that they can do so with the artificial networks presented here.

We must observe in general that for a given number of vertices, when the density is specified at 0.9, the distribution charts presented seem to have almost no difference between the variation in number of communities. This gives way to the possibility that each of those two pair of classes might not present enough differences when reduced to the measures under study, that would allow them to be differentiated.

5.4 Learning to Classify Artificial Networks

With the instances created and their statistics reviewed, it is time to put these artificial networks under the scrutiny of the classification algorithms, to evaluate their learning abilities.

In order to ensure a consistent and reproducible environment, where all of the proposed classification algorithms are available, we will make use of **Keel**.

Definition 21. ***KEEL (Knowledge Extraction based on Evolutionary Learning)****[6] is an open source Java software tool that can be used for a large number of different knowledge data discovery tasks. It is freely available at* `http://www.keel.es/`.

With a software platform as Keel, we can easily build experiments by providing the instances of the classification problem described, then ask it to apply different algorithms. All of the classification techniques in Section 3.3 are included.

The instances presented to the classification algorithms must contain all the information of the complexity measures computed for each network in the different domains/classes. To achieve this, we will convert each network to a row in the input files. Each row will indicate the value for each of the measures computed for the given network, and the assigned class/domain at the end. With a list of all such instance rows, we can proceed to use Keel as a platform for executing the classification algorithms.

Since some of the implementations proposed make use of a seed for pseudo-random behavior, the results presented will be an average of 250 successive runs of each algorithm. To analyze the statistical significance of the results obtained, we will also provide a histogram for the testing accuracy observed and compare it to either (1) a uniform distribution (for non-stochastic results) or (2) an equivalent normal distribution according to its mean and standard deviation (for stochastic results).

Because the objective of this work is to confirm the plausibility of applying classification techniques to learn about networks via their complexity measures, we will restrict the experiments to the simpler variants of the algorithms reviewed. Keel provides a wide range of more complex implementations that might allow to improve on our results, as well as other classification algorithms that might also be suited for the experiments. Such a task is a possible focus of future research.

From the algorithms offered, we propose to create the following experiments, listed in Table 5.2.

The parameters used for each algorithm are left at their defaults as proposed by Keel. They are listed next in Table 5.3.

To build the training and testing sets, we will apply Keel's experiment building feature. The 20 instances for each of the 8 artificial classes provide

Table 5.2: Keel algorithms to classify our CiGRAM networks

Full Name	Short Name	Reference
K-Nearest Neighbors Classifier	KNN-C	[47]
Improved KNN Condensation	IKNN-C	[205]
C-SVM	C_SVM-C	[45]
NU-SVM	NU_SVM-C	[172]
Bioinformatics-oriented hierarchical evolutionary learning	BioHel-C	[9]

Table 5.3: Parameters used for the Keel algorithms

Algorithm	Parameter	Value
KNN-C	K Value	1
	Distance Function	Euclidean
IKNN-C	Gamma rate with respect to the number of instances of the minority class	0.1
	Xi multiplicative factor	0.2
	Xi exponential factor	−0.5
	Distance Function	Euclidean
C_SVM	KERNELtype	RBF
	C	100.0
	eps	0.001
	degree	1
	gamma	0.01
	coef0	0.0
	nu	0.1
	p	1.0
	shrinking	1
NU_SVM	KERNELtype	RBF
	C	1000.0
	eps	0.001
	degree	10
	gamma	0.01
	coef0	0.0
	nu	0.1
	p	1.0
	shrinking	1

Algorithm	Parameter	Value
BioHEL-C	defClass	disabled
	popSize	100
	selectAlg	tournament WOR
	tournamentSize	1
	crossoverProbability	1.6
	mutationProbability	1.6
	elitismEnabled	TRUE
	numIterations	100
	numRepetitions Learning	1
	generalizeProbability	1.1
	specializeProbability	1.1
	winMethod	ilas
	numStrataWindowing	1
	numStages	1
	smartInitMethod	TRUE
	classWiseInit	TRUE
	probOne	1.75
	useMDL	TRUE
	numIterationsMDL	10
	initialTheory LengthRatio	1.01
	mdlWeightRelaxFactor	1.9
	coverageBreakpoint	1.1
	coverageRatio	1.9

us with a total of 160 collection of inputs. The training set will be built by taking 15 of the 20 instances for each class, while the remaining 5 will be grouped into a testing set.

One of the ways in which we can observe the classification capabilities of the different algorithms is using the confusion matrices obtained after running each of them. Each cell (r, c) of the matrix presents the percentage of instances of the class determined by row r that were assigned the class specified by column c. To simplify reading the results, the classes are presented with the simplified names shown in Table 5.4, keeping the order in which they were presented in the statistical distribution charts. The next section will then provide a summary of the exact values for classification

accuracy.

Table 5.4: Simplified names for the artificial classes

Full Name	Short Name
n100-d0.1-k10	Class1
n100-d0.1-k50	Class2
n100-d0.9-k10	Class3
n100-d0.9-k50	Class4
n1000-d0.1-k10	Class5
n1000-d0.1-k50	Class6
n1000-d0.9-k10	Class7
n1000-d0.9-k50	Class8

Table 5.5: Training Set Confusion Matrix for KNN-C for CiGRAM networks

	Class1	Class2	Class3	Class4	Class5	Class6	Class7	Class8
Class1	100%	0%	0%	0%	0%	0%	0%	0%
Class2	0%	100%	0%	0%	0%	0%	0%	0%
Class3	0%	0%	33%	67%	0%	0%	0%	0%
Class4	0%	0%	47%	53%	0%	0%	0%	0%
Class5	0%	0%	0%	0%	100%	0%	0%	0%
Class6	0%	0%	0%	0%	0%	100%	0%	0%
Class7	0%	0%	0%	0%	0%	0%	87%	13%
Class8	0%	0%	0%	0%	0%	0%	20%	80%

Table 5.6: Testing Set Confusion Matrix for KNN-C for CiGRAM networks

	Class1	Class2	Class3	Class4	Class5	Class6	Class7	Class8
Class1	100%	0%	0%	0%	0%	0%	0%	0%
Class2	0%	100%	0%	0%	0%	0%	0%	0%
Class3	0%	0%	20%	80%	0%	0%	0%	0%
Class4	0%	0%	60%	40%	0%	0%	0%	0%
Class5	0%	0%	0%	0%	100%	0%	0%	0%
Class6	0%	0%	0%	0%	0%	100%	0%	0%
Class7	0%	0%	0%	0%	0%	0%	100%	0%
Class8	0%	0%	0%	0%	0%	0%	60%	40%

Table 5.5 shows the confusion matrix for the training set of the *KNN-C* algorithm, while table 5.6 presents the values for the testing set. They present a promising scenario, since there is an mix up when classifying

*Class*3 and *Class*4, and another between *Class*7 and *Class*8. This difference is more significant in the training than in the testing set of instances, most likely due to the higher number of samples in the first case which allow a more precise result. In general terms, it seems that the classification has been possible.

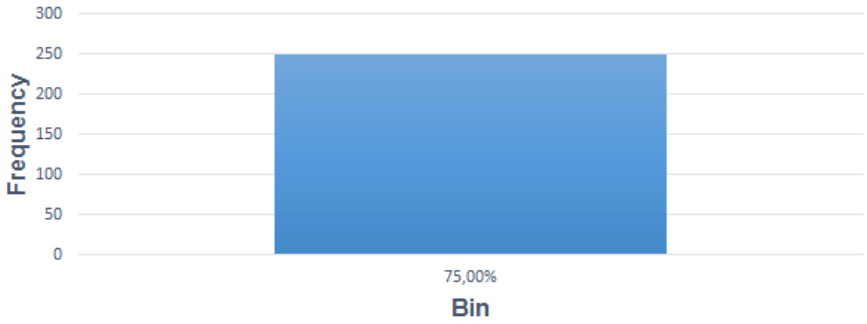


Figure 5.16: Histogram for KNN-C testing accuracies of CiGRAM networks. A uniform result is produced due to the lack of pseudo-random numbers in the algorithm.

The testing accuracy distribution histogram is presented in Figure 5.16. Here we can confirm a uniform distribution of the values, since the algorithm does not make use of pseudo-random numbers.

Table 5.7: Training Set Confusion Matrix for IKNN-C for CiGRAM networks

	Class1	Class2	Class3	Class4	Class5	Class6	Class7	Class8
Class1	87%	13%	0%	0%	0%	0%	0%	0%
Class2	7%	93%	0%	0%	0%	0%	0%	0%
Class3	0%	0%	33%	67%	0%	0%	0%	0%
Class4	0%	0%	53%	47%	0%	0%	0%	0%
Class5	0%	0%	0%	0%	93%	0%	0%	7%
Class6	0%	0%	0%	0%	100%	0%	0%	0%
Class7	0%	0%	0%	0%	0%	0%	53%	47%
Class8	0%	0%	0%	0%	0%	0%	33%	67%

In Tables 5.7 and 5.8 the training and testing results for the *IKNN-C* variant are presented. This algorithm shows similar values, but a reduction in the accuracy appears due to the complete missclassification of *Class*6 instances as belonging to *Class*5. Also, both training and testing sets present some missclassifications between *Class*1 and *Class*2.

Table 5.8: Testing Set Confusion Matrix for IKNN-C for CiGRAM networks

	Class1	Class2	Class3	Class4	Class5	Class6	Class7	Class8
Class1	80%	20%	0%	0%	0%	0%	0%	0%
Class2	0%	100%	0%	0%	0%	0%	0%	0%
Class3	0%	0%	40%	60%	0%	0%	0%	0%
Class4	0%	0%	60%	40%	0%	0%	0%	0%
Class5	0%	0%	0%	0%	100%	0%	0%	0%
Class6	0%	0%	0%	0%	100%	0%	0%	0%
Class7	0%	0%	0%	0%	0%	0%	100%	0%
Class8	0%	0%	0%	0%	0%	0%	60%	40%

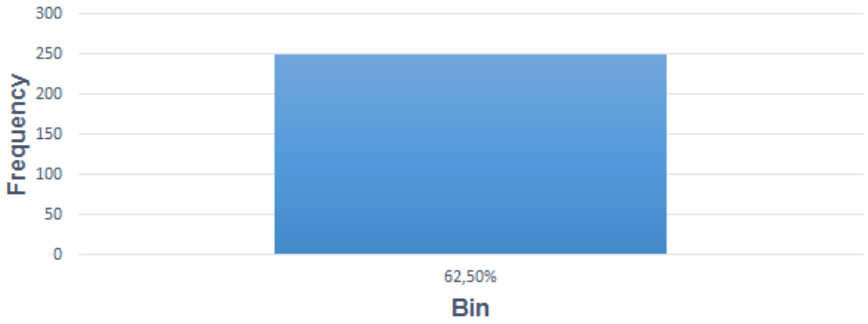


Figure 5.17: Histogram for IKNN-C testing accuracies of CiGRAM networks. Since all runs provided the same result, the observed values present a uniform distribution.

Figure 5.17 confirms that this algorithm is also deterministic in its results.

Table 5.9: Training Set Confusion Matrix for C_SVM-C for CiGRAM networks

	Class1	Class2	Class3	Class4	Class5	Class6	Class7	Class8
Class1	100%	0%	0%	0%	0%	0%	0%	0%
Class2	0%	100%	0%	0%	0%	0%	0%	0%
Class3	0%	0%	93%	7%	0%	0%	0%	0%
Class4	0%	0%	67%	33%	0%	0%	0%	0%
Class5	0%	0%	0%	0%	100%	0%	0%	0%
Class6	0%	0%	0%	0%	0%	100%	0%	0%
Class7	0%	0%	0%	0%	0%	0%	87%	13%
Class8	0%	0%	0%	0%	0%	0%	27%	73%

When applying the Support Vector Machine implementation present in

Table 5.10: Testing Set Confusion Matrix for C_SVM-C for CiGRAM networks

	Class1	Class2	Class3	Class4	Class5	Class6	Class7	Class8
Class1	100%	0%	0%	0%	0%	0%	0%	0%
Class2	20%	80%	0%	0%	0%	0%	0%	0%
Class3	0%	0%	100%	0%	0%	0%	0%	0%
Class4	0%	0%	80%	20%	0%	0%	0%	0%
Class5	0%	0%	0%	0%	100%	0%	0%	0%
Class6	0%	0%	0%	0%	0%	100%	0%	0%
Class7	0%	0%	0%	0%	0%	0%	80%	20%
Class8	0%	0%	0%	0%	0%	0%	40%	60%

Keel as *C_SVM-C* we obtain the training and testing outputs displayed in tables 5.9 and 5.10 respectively. It shows a stable result, comparable to that observed for the first algorithm. There is again a partial misclassification between $Class3$ and $Class4$, and there is also some confusion between $Class7$ and $Class8$. It is convenient to remember that the classes were generated with a simple scheme that was not intended to provide any specific complication for the algorithms. Moreover, the fact that all networks are compressed to the same number of measures means that it is not necessarily more difficult to differentiate between small networks that it is larger ones.

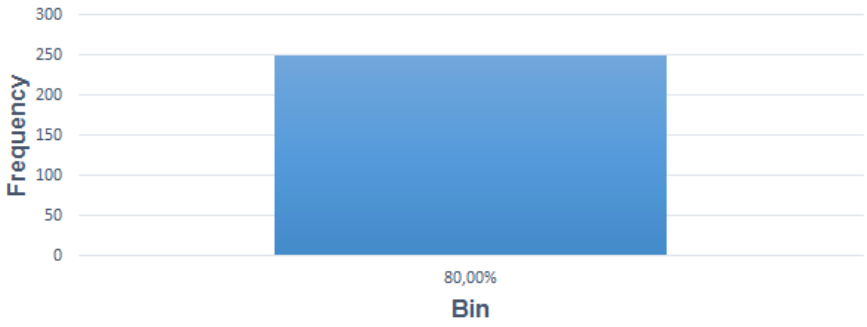


Figure 5.18: Histogram for C_SVM-C testing accuracies of CiGRAM networks

Although the algorithm, as offered in Keel, provides the possibility of indicating a random seed, we can observe in Figure 5.18 that the results obtained do no deviate from one single outcome, ensuring a statistical significant value.

In order to analyze a variant of Support Vector Machines, *NU_SVM-C* is considered next. Table 5.11 contains the confusion matrix for the training set and Table 5.12 the one for the testing set. The outcome is not that dif-

Table 5.11: Training Set Confusion Matrix for NU_SVM-C for CiGRAM networks

	Class1	Class2	Class3	Class4	Class5	Class6	Class7	Class8
Class1	100%	0%	0%	0%	0%	0%	0%	0%
Class2	0%	100%	0%	0%	0%	0%	0%	0%
Class3	0%	0%	60%	40%	0%	0%	0%	0%
Class4	0%	0%	47%	53%	0%	0%	0%	0%
Class5	0%	0%	0%	0%	100%	0%	0%	0%
Class6	0%	0%	0%	0%	0%	100%	0%	0%
Class7	0%	0%	0%	0%	0%	0%	20%	80%
Class8	0%	0%	0%	0%	0%	0%	7%	93%

Table 5.12: Testing Set Confusion Matrix for NU_SVM-C for CiGRAM networks

	Class1	Class2	Class3	Class4	Class5	Class6	Class7	Class8
Class1	100%	0%	0%	0%	0%	0%	0%	0%
Class2	20%	80%	0%	0%	0%	0%	0%	0%
Class3	0%	0%	80%	20%	0%	0%	0%	0%
Class4	0%	0%	60%	40%	0%	0%	0%	0%
Class5	0%	0%	0%	0%	100%	0%	0%	0%
Class6	0%	0%	0%	0%	0%	100%	0%	0%
Class7	0%	0%	0%	0%	0%	0%	80%	20%
Class8	0%	0%	0%	0%	0%	0%	20%	80%

ferent from the basic implementation, most likely because the improvement in this version is focused on allowing a finer tuning of the situation, and we are explicitly avoiding such customization for now, running only with the default values included in Keel.

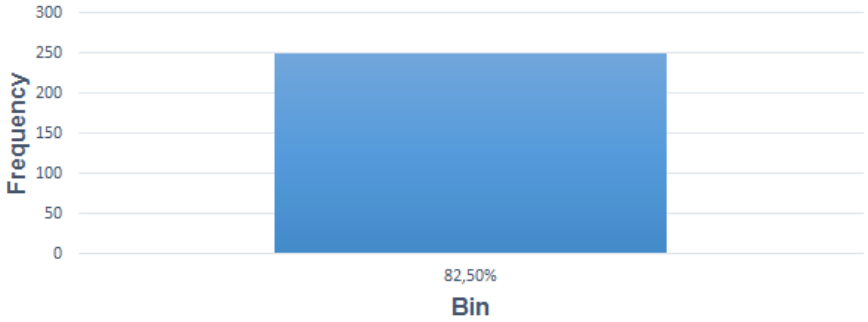


Figure 5.19: Histogram for NU_SVM-C testing accuracies of CiGRAM networks

As for the previous implementation, the outcome presented in Figure

5.19 is that of a constant result across all runs of the experiment.

Table 5.13: Training Set Confusion Matrix for BioHEL-C for CiGRAM networks

	Class1	Class2	Class3	Class4	Class5	Class6	Class7	Class8
Class1	100%	0%	0%	0%	0%	0%	0%	0%
Class2	0%	100%	0%	0%	0%	0%	0%	0%
Class3	0%	0%	91%	9%	0%	0%	0%	0%
Class4	0%	0%	12%	88%	0%	0%	0%	0%
Class5	0%	0%	0%	0%	100%	0%	0%	0%
Class6	0%	0%	0%	0%	0%	100%	0%	0%
Class7	0%	0%	0%	0%	0%	0%	98%	2%
Class8	0%	0%	0%	0%	0%	0%	1%	99%

Table 5.14: Testing Set Confusion Matrix for BioHEL-C for CiGRAM networks

	Class1	Class2	Class3	Class4	Class5	Class6	Class7	Class8
Class1	96%	2%	1%	1%	0%	0%	0%	0%
Class2	20%	80%	0%	0%	0%	0%	0%	0%
Class3	0%	0%	55%	45%	0%	0%	0%	0%
Class4	0%	0%	41%	59%	0%	0%	0%	0%
Class5	0%	0%	0%	0%	98%	2%	0%	0%
Class6	0%	0%	0%	0%	10%	90%	0%	0%
Class7	0%	0%	0%	0%	0%	0%	83%	16%
Class8	2%	0%	0%	0%	0%	0%	23%	75%

Finally, we are left with Tables 5.13 and 5.14, which show the accuracy when classifying instances in the training and testing sets while running *BioHEL-C*. Although the training set presents a significant improvement over the results observed when applying the other algorithms, we see that the testing set still creates some confusion. We must remember here that the values presented are an average over 250 runs, where both higher and lower classification performances were observed.

In Figure 5.20 a histogram is presented with the different values obtained for the testing accuracy of the repeated experimental attempts. The chart also presents an overlap simulation of a normal distribution with the equivalent mean and standard deviation observed (79.48% and 4.47% respectively).

Despite the slight misclassifications, we must point out that with the right tool, and the correct parameters, it has been possible to obtain a good classification accuracy with all of the algorithm families proposed.

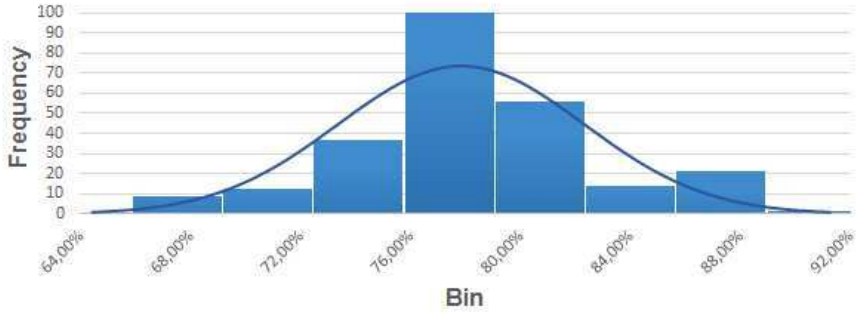


Figure 5.20: Histogram for BioHEL-C testing accuracies of CiGRAM networks

The classes that cause more degradation in the overall accuracy are $Class3$, which is confused with $Class4$, and $Class7$, often mixed with $Class8$. By reviewing Table 5.4, we can reinterpret these results. $n100 - d0.9 - k10$ gets confused with $n100 - d0.9 - k50$ and $n1000 - d0.9 - k10$ with $n1000 - d0.9 - k50$. These means that the algorithms are having trouble in differentiating networks with different number of communities when their density is higher. This comes as no surprise when comparing the results obtained with the statistical distribution charts presented in Section 5.3, where these pairs were presented on the same range of values for all the measures under study, even presenting only a very compact quartile box, which means that most networks in these classes gave similar values.

5.5 Results for the Artificial Networks

After considering all the tables presented in the previous section, we can now build a detail of the observed accuracy for each algorithm. The P-value presented was computed using a Friedman test. Table 5.15 shows these values. We have already observed that *KNN-C* and *IKNN-C* show no classification ability with the instances provided, that *C_SVM-C* and *NU_SVM-C* share similar values in both training and testing accuracy, and that *BioHEL-C* has an even higher accuracy when training, but no improvement over the Support Vector Machine algorithms when considering its testing accuracy.

With these results, we can be confident in continuing the proposed ideas of classifying networks by means of their complexity measures, this time applying the algorithms to the real networks researched and presented in Section 4.2.

One final detail that is of interest with the results produced by the

Table 5.15: Classification Accuracy for the Artificial Networks

	KNN-C	IKNN-C	C_SVM-C	NU_SVM-C	BioHEL-C
Training Set	81.67%	59.17%	85.33%	78.33%	96.89%
Testing Set	75.00%	62.50%	80.00%	82.50%	79.48%

P-value is 3.57×10^{-215} for the training set and 1.71×10^{-179} for the testing set.

Table 5.16: Rules produced by BioHEL for the Artificial Networks

```
0:Betweenness(Q2) is > 0.20975111
  THEN n1000-d0.1-k50 (Class6)

1:Betweenness(IQR) is > 0.14870113
  THEN n1000-d0.1-k10 (Class5)

2:Clustering(Q1) is < 0.5174123
  OR PageRank(Max) is < 0.95674664
  THEN n100-d0.1-k50 (Class2)

3:Eigenvector(Q2) is < 0.743402
  THEN n100-d0.1-k10 (Class1)

4:OffdiagonalComplexity is < 0.83688915
  OR Clustering(IQR) is > 0.091581926
  OR PageRank(Avg) is < 0.9197471
  THEN n1000-d0.9-k50 (Class8)

5:|E| is > 0.01002797
  OR Closeness(Q2) is > 0.38967088
  THEN n1000-d0.9-k10 (Class7)

6:|V| is > 0.40778017
  OR Codelength is > 0.63766336
  OR Betweenness(Avg) is < 0.5914795
  OR Betweenness(Q1) is
    BETWEEN (0.048583686 AND 0.38698635)
  OR Betweenness(IQR) is < 0.727874
  OR MaximumDegree is > 0.70464796
  OR OffdiagonalComplexity is > 0.37859428
  OR Clustering(IQR) is < 0.5421958
  OR Clustering(Max) is > 0.30774963
  OR Eigenvector(Q1) is > 0.55717087
  OR Eigenvector(IQR) is
    BETWEEN (0.63593376 AND 0.90244687)
  OR PageRank(Avg) is < 0.7112693
  OR PageRank(Q1) is < 0.4935823
  OR PageRank(Q3) is < 0.4277151
  OR PageRank(Min) is < 0.4896802
  THEN n1000-d0.9-k50 (Class8)

7:Closeness(Max) is
    BETWEEN (0.2988854 AND 0.97010934)
  OR OffdiagonalComplexity is < 0.49941477
  OR Clustering(IQR) is < 0.96575373
  OR Eigenvector(Q1) is > 0.7926535
  OR Eigenvector(Q2) is
    BETWEEN(0.79922175 AND 0.967829)
  OR PageRank(Q2) is < 0.690705
  THEN n100-d0.9-k50 (Class4)

8:OffdiagonalComplexity is > 0.45171407
  THEN n100-d0.9-k10 (Class3)

9:ELSE n100-d0.9-k50 (Class4)
```

algorithms is the mentioned structure of BioHEL output, which takes the form of a set of rules that can be then used to understand the "logic" behind the classification. In the case of our artificial networks, the best rule set generated obtained a training accuracy of 98.33% and a testing accuracy of 92.50%, and produced the rules presented in Table 5.16.

When analyzing these rules, it can be seen that BioHEL ended up using the most complex rules to try and differentiate between $Class3$ and $Class4$, also applied many different measures when classifying $Class4$ and $Class8$, and used the default rule to choose $Class4$ in all other situations not already considered.

This classification criteria is not only human-readable, but also easy to parse and insert into an automatic program that could classify instances just by reading the values in the input. Furthermore, given that repeated experiment attempts provide each with one resulting rule set, they can be combined in a majority voting scheme, possibly granting increase classification accuracy.

5.6 Review

Throughout this chapter we have presented CiGRAM, an artificial network generator, that allowed us to define a set of 8 evenly distributed network classes, with a distribution similar to that obtained for the real networks in Section 4.2.

The networks created were then measured using the different formulas determined in Section 4.3, and the distribution of their statistics was presented. Some of the classes created were observed to have a greater overlap in the values obtained, while others seemed to be clearly distinguishable.

With the set of complexity measures computed for each of the networks, the classification algorithms suggested in Section 3.3 were run and the results presented. We used the Keel framework for generating the experiments, and the parameters for each algorithm were detailed.

After reviewing the experimental results, we could confirm that the classes with more overlap in the complexity measures were more difficult to classify, but despite that most of the classification algorithms employed obtained a testing accuracy around 80%. We will analyze in the following chapter whether that level of accuracy can be maintained when the domains and networks analyzed come from the real world examples described in Section 4.2.

Capítulo 5: Validando la habilidad de clasificación

Este capítulo presenta un conjunto de 160 redes generadas artificialmente y clasificadas en 8 dominios, en base a su criterio de construcción. De esta manera se pretende validar la idea de que es posible clasificar redes en distintas categorías de acuerdo a los valores observados para las métricas elegidas.

Para obtener las redes artificiales utilizamos el generador CiGRAM, que permite especificar ciertos parámetros deseables de las redes a crear.

Para todas las redes generadas se calculan las métricas y se presenta su distribución estadística en la sección 5.3. Podemos observar que para algunas de las métricas los valores obtenidos se solapan en los distintos dominios (por ejemplo PageRank en la figura 5.15). Sin embargo, en otros casos la diferencia es más notoria, como se observa para Codelength en la figura 5.8.

En la sección 5.4 se presenta el resultado de ejecutar los algoritmos de clasificación con un conjunto determinado de parámetros. Para garantizar que los experimentos sean comparables, todos son ejecutados a través del ambiente de experimentación Keel [6].

Luego de ejecutar los algoritmos de clasficación, pudimos comprobar que los dominios que presentaban mayor solapamiento en la distribución estadística de sus valores fueron los que presentaron menor precisión en los resultados. A pesar de ello, la mayoría de los algoritmos elegidos consiguió llegar a un rendimiento del 80%.

En el próximo capítulo intentaremos estudiar si estos niveles de precisión se mantienen al trabajar con las redes reales relevadas en el capítulo 4 y sus correspondientes dominios.

Chapter 6

Experiments

6.1 Description

This chapter defines a set of experiments planned to analyze the feasibility of solving the classification problem using the information described up to now. After each experiment we present the results obtained, and an analysis of the observations.

The first aim will be to reproduce the classification experiments using the learning algorithms included in Keel, presented in Chapter 5. Only this time, the categories and instances to consider will be the real world networks presented in Section 4.2.

As mentioned in Section 5.4, the different networks will be converted to instances of our training and testing sets. The input files for Keel contain, for each network, the set of measurement values represented by real numbers between 0 and 1 (since we have normalized all results as explained in 4.3.3), and the name of the domain. The detection and handling of outlier values is described in the current chapter in Section 6.3.

Our final objective is the confirmation of our hypothesis: that studying the complexity measures of networks from different domain problems allows to find defining/discriminating qualities for each category. Moreover, we want to find a deterministic set of steps using classification algorithms to ensure a reproducible way of harnessing this findings. Although the previous chapter provided encouraging results, with testing accuracies over 80% in most cases, and also individual results over 90%, we need to confirm that the level of success for real networks classified according to the domains proposed is not reduced.

Each experiment implies selecting a subset of the instances available and running the classification algorithms with a specific set of measures from those presented. The set of classification algorithms is the same as presented in Table 5.2. The parameters assigned for each of them is kept at its default values, presented in Table 5.3. The training and testing sets are once again input files, where each contains a row (instance) for each network considered in the set. The values for an instance are the sequence of real numbers between 0 and 1 that represent the complexity measure values computed for the network, finishing with a string value that represents the domain for the network. All results presented are averaged over 250 runs, with a statistical analysis of the testing accuracy results to ensure its significance.

In order to avoid cluttering this chapter with tables, only the average training and testing accuracies are presented, while the confusion matrices for each experiment are presented in the appendices of this document. The histogram distribution is only presented for *BioHEL* results, since the other classification algorithms proved to provide deterministic results for each given experiment, and would imply presenting a repetition of single bin histograms.

6.2 Direct classification

The first approach will be to apply the same algorithms from Keel shown before, considering all instances and categories without additional processing. The next sections will then consider additional possible improvements on the experiment set up.

The training and testing set will first be built using Keel's K-fold building feature, selecting a 10-fold cross validation scheme. Afterwards we will perform additional experiments selecting a specific set of the networks as representatives for each domain.

We expect these trials to include some uneven results as the instances chosen for each run are selected at random.

Table 6.1: Classification accuracy when considering all real networks reviewed

	KNN-C	IKNN-C	C_SVM-C	NU_SVM-C	BioHEL-C
Training Set	66.51%	26.30%	76.48%	80.56%	93.45%
Testing Set	66.25%	36.67%	65.42%	62.92%	62.21%

P-value is 3.57×10^{-215} for the training set and 9.52×10^{-120} for the testing set.

Table 6.1 shows that the *IKNN-C* algorithm is the one having the most trouble classifying the instances, even failing to obtain an accurate result for the training set. All other techniques show a reduced testing accuracy when compared to the attempt at classifying artificial instances as presented in Section 5.5. It is interesting to note that only *KNN-C* obtains a comparable training and testing accuracy, possibly hinting at an over-specialization of the other algorithms during their training.

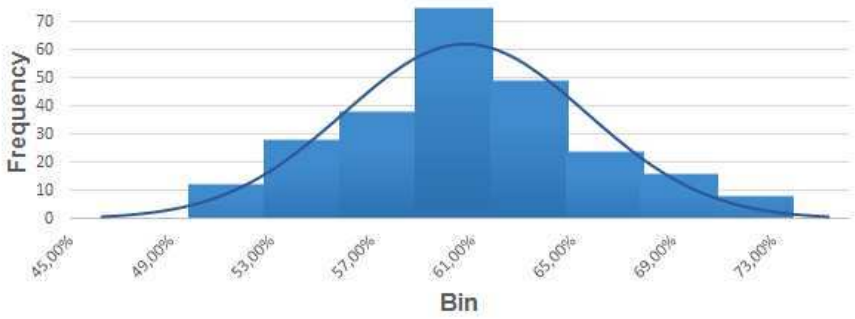


Figure 6.1: Histogram for BioHEL-C testing accuracies when considering all real networks reviewed, using Keel automatic 10-fold

Figure 6.1 contains the distribution of results for the training accuracies observed, with a standard deviation of 4.82%. The maximum result achieved by this algorithm in the experiment was 75.00%.

At this point it is interesting to observe that the correlation observed for the real networks (Figure 4.23), which was significantly lower than that of the artificial networks (Figure 5.5), was not a good predictor of the classification accuracy in the experiments proposed. Contrary to what the correlations could have implied, it seems the algorithms find it easier to discriminate between the artificial categories than the real ones.

On the other hand, the box plot figures presented in Section 4.3.3 for the real networks and 5.3 for the artificial networks, do serve as good predictors for the observed classification accuracy. If the differences in the different domains are more closely related to the classification capability of the algorithms, we can propose the following hypothesis: a rule generating process such as the one included in BioHEL should be able to produce, for an experiment with statistics that present a *clearer distinction between the categories* (as seen for the artificial networks) a *smaller rule set*, with rules that make use of *less attributes*. For an experiment where the statistics present a *less clear distinction* (as seen for the real networks), we would

expect on average a *larger rule set*, where each rule would generally require a *larger number of attributes* to try to discriminate between the categories.

To test this hypothesis, we present in Table 6.2, the average and standard deviation for the number of rules produced by BioHEL in the experiments that analyze the artificial networks, and compare it to the same value when considering the real networks. We also present the average and standard deviation of the number of attributes used per rule in the rule set.

Table 6.2: Number of rules and number of attributes per rule, produced by BioHEL in the artificial and real networks experiments.

	Rules per rule set		Attributes per rule	
	Average	St. Dev.	Average	St. Dev.
Artificial Networks	8.74	1.04	2.35	0.74
Real Networks	18.51	1.93	4.61	0.88

The values observed in Table 6.2 confirm our hypothesis, since the real networks present a rule set that is on average more than twice as long and with double the number of attributes required per rule.

We continue now with additional experimental proposals, with the objective of trying to improve the classification accuracy observed.

6.3 Outlier detection

One possible source of noise in the classification results is generated by the existence of outliers in the data.

Definition 22. *An **outlier** is any instance in a collection of data whose information is not aligned with the general range of values under study. This could be evidence of an error in the gathering of the data, or a measurement that represents a situation with a lower chance of occurring.*

Since classification techniques rely in part on the established range of values for the measurements under study, we decided to run additional experiments where possible outliers in the instances researched are detected and singled out. They will be separated in the following experiments from both training and testing sets, to be reused later on as additional validation for the algorithm solutions.

To provide a simple and effective detection of outliers, we will apply the technique presented by John Tukey in [196]. The method implies considering the range of values presented by all the data, computing its First and

Third Quartiles, as well as the Inter-Quartile Range, and then determining a lower and upper bound:

$$Lower\ Bound = (1^{st}\ \text{Quartile}) - (1.5 \times \text{Inter-Quartile Range}) \quad (6.1)$$

$$Upper\ Bound = (3^{rd}\ \text{Quartile}) + (1.5 \times \text{Inter-Quartile Range}) \quad (6.2)$$

Since our instances are composed of several measurements, and any of them could be used to determine an outlier, we will determine the bound for $|V|$ and $|E|$ for each network domain. Then, we will proceed to classify as outliers any instance where these measurements are outside the lower or upper bound in their respective domains, for any of the two measures considered.

After applying the technique mentioned, the networks listed in Table 6.3 were categorized as outliers and removed from all experiments that follow in the chapter, unless explicitly stated.

Table 6.3: Outlier instances

Network	Domain
NT-bio-dmela	Assorted Biological
Github Profiles	Code
Astro Physics	Collaboration
Condensed Matter	Collaboration
High Energy Physics	Collaboration
Enron Mail	Communication
European Mail	Communication
as-caida2007	Infrastructure
internet-as	Infrastructure
Internet Routers	Infrastructure
p2p-gnutella	Infrastructure
BIOMD0000000014-RDG	Reaction Dependency Graph
BIOMD0000000183-RDG	Reaction Dependency Graph
BIOMD0000000255-RDG	Reaction Dependency Graph
BIOMD0000000014-SNG	Species Network Graph
BIOMD0000000183-SNG	Species Network Graph
BIOMD0000000255-SNG	Species Network Graph
NT-rt-pol	Tweet

6.4 All instances except outliers

Once the detailed outliers were removed, the correlation for the different measures was computed again.

	|V|	|E|	CL	Diam	PL	Betw	Close	GE	MD	OdC	Clust	Eigen	PR
|V|	1,00	0,67	0,52	0,19	0,14	0,61	-0,16	0,23	0,38	0,46	-0,20	-0,29	-0,25
|E|	0,67	1,00	0,57	0,16	0,11	0,58	-0,01	0,02	0,24	0,54	-0,05	-0,22	-0,19
CL	0,52	0,57	1,00	0,54	0,48	0,63	0,34	-0,04	0,27	0,72	-0,14	-0,30	-0,58
Diam	0,19	0,16	0,54	1,00	0,89	0,48	0,69	-0,07	-0,01	0,38	-0,38	-0,43	-0,48
PL	0,14	0,11	0,48	0,89	1,00	0,46	0,70	-0,18	-0,11	0,26	-0,38	-0,20	-0,43
Betw	0,61	0,58	0,63	0,48	0,46	1,00	0,34	-0,14	0,19	0,43	-0,21	-0,31	-0,25
Close	-0,16	-0,01	0,34	0,69	0,70	0,34	1,00	-0,35	-0,22	0,13	-0,11	-0,19	-0,20
GE	0,23	0,02	-0,04	-0,07	-0,18	-0,14	-0,35	1,00	0,06	0,23	-0,19	-0,18	-0,17
MD	0,38	0,24	0,27	-0,01	-0,11	0,19	-0,22	0,06	1,00	0,06	-0,14	-0,20	-0,18
OdC	0,46	0,54	0,72	0,38	0,26	0,43	0,13	0,23	0,06	1,00	0,06	-0,56	-0,49
Clust	-0,20	-0,05	-0,14	-0,38	-0,38	-0,21	-0,11	-0,19	-0,14	0,06	1,00	0,18	0,19
Eigen	-0,29	-0,22	-0,30	-0,43	-0,20	-0,31	-0,19	-0,18	-0,20	-0,56	0,18	1,00	0,30
PR	-0,25	-0,19	-0,58	-0,48	-0,43	-0,25	-0,20	-0,17	-0,18	-0,49	0,19	0,30	1,00

Figure 6.2: Correlation between the selected measures when the outliers are removed

Figure 6.2 presents the new correlations observed. When compared to the previous values in Figure 4.23 from Section 4.3.1, the most significant change is a slight reduction of some higher correlations.

The next experiment will be then to run the algorithms, again using Keel partition generation (10 folds), this time considering the instances that are not outliers. We expect an equal or better performance from all the classification techniques used, although it might still be possible to encounter a lower testing accuracy, considering that the partitioning of the sets is done without any specific considerations.

Table 6.4: Classification accuracy when not considering outliers (automatic partitioning)

	KNN-C	IKNN-C	C_SVM-C	NU_SVM-C	BioHEL-C
Training Set	66.97%	27.47%	76.02%	81.38%	95.27%
Testing Set	67.55%	38.79%	64.84%	61.78%	64.31%

P-value is 3.57×10^{-215} for the training set and 2.68×10^{-113} for the testing set.

Table 6.4 presents little to no change with respect to the previous experimental result. There is some small increase in training and testing accuracy for most algorithms, but no testing result surpasses a 70% accuracy. The *IKNN-C* algorithm continues to present the lowest performance, with no domain distinguished from another. The confusion matrices for both training and testing accuracies are presented in the appendices.

The histogram for the BioHEL testing accuracies is present in Figure 6.3. The standard deviation observed in the normal curve is 5.56%. The maximum value obtained is 75.91%, completing the situation of slight improvements over the previous experiment.

Figure 6.3: Histogram for BioHEL-C testing accuracies when discarding outliers, using Keel automatic 10-fold

Table 6.5: Number of rules and number of attributes per rule, produced by BioHEL when discarding outliers, using Keel automatic 10-fold

Rules per rule set		Attributes per rule	
Average	St. Dev.	Average	St. Dev.
18.05	1.69	4.65	0.93

In Table 6.5 we confirm that the average number of rules and attributes per rule in the BioHEL results is still presenting a complexity far greater than that observed for the artificial networks.

Beginning with the next proposed experiment, we aim at creating a better distribution of the networks (instances), in order to provide a more balanced data set. It is our expectation that with a better distribution of the training and testing sets in regards to domain representation, the classification algorithms can run in a scenario that allows for improved accuracies.

6.5 Manual partitioning of the training/testing sets

Continuing with the experimental design, we will propose taking a specific 70% of the instances and use them in a training set, with the remaining 30% employed as the testing set.

To select them, we will sort the instances in each domain by "size" and roughly 1 for every 3 to obtain the chosen ratio. To allow for a wider range of coverage in our experiments, size will be determined using the number of vertices, and then another experiment will consider the number of edges.

This implies we will be able to run 2 new experiments, to which we will refer to in the following chapter as "Vertices 70/30" and "Edges 70/30".

Afterwards, the same process will be repeated to create a partition where 90% of the instances are used in the training set, and 10% left for the testing set. Again, we will manually select 1 for every 10 instances to achieve this partitioning. The resulting proposals will be presented then as "Vertices 90/10" and "Edges 90/10".

Our expectation for these experiments is that they should provide a more accurate classification, since the sampling of the training set has been specifically chosen to be representative of the complete set, with all of the testing values well within the same range.

The detail of the instances chosen for each set can be found in the appendices of this document.

Table 6.6: No Outliers Classification Accuracy (70% / 30% manual partitioning based on $|V|$)

	KNN-C	IKNN-C	C_SVM-C	NU_SVM-C	BioHEL-C
Training Set	64.90%	22.52%	81.46%	81.46%	96.25%
Testing Set	61.97%	25.35%	60.56%	59.15%	59.94%

P-value is 2.44×10^{-204} for the training set and 2.44×10^{-167} for the testing set.

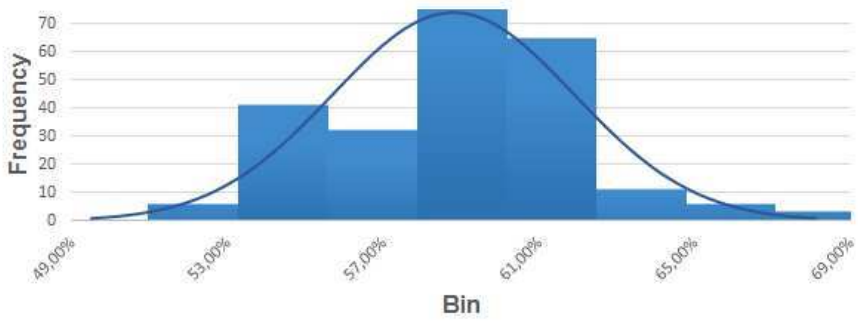


Figure 6.4: Histogram for BioHEL-C testing accuracies when discarding outliers, using 70% / 30% manual partitioning based on $|V|$

Table 6.6 displays experimental accuracies when manually building the training sets with 70% of the instances, leaving 30% for testing, according to number of vertices. The testing accuracies obtained are even less than for previous experiments, reducing the possibilities of applying this approach as a means of improving the classification ability of the algorithms.

Figure 6.4 presents the histogram when applying BioHEL, with an observed standard deviation of 3.10%, and an overall maximum of 69.01%. This chart continues to prove that the results obtained are consistent across several runs, although in this scenario even the maximum obtained fails to represent a useful classification scheme.

Table 6.7: Number of rules and number of attributes per rule, produced by BioHEL when discarding outliers, using 70% / 30% manual partitioning based on $|V|$

Rules per rule set		Attributes per rule	
Average	St. Dev.	Average	St. Dev.
16.42	1.74	4.06	0.88

Table 6.7 informs the average and standard deviation for number of rules per rule set and number of attributes per rule in the BioHEL results. Although there is a slight decrease in the number of rules, it is still double the value observed previously for the artificial networks, and the number of attributes per rule remains at the same levels.

Table 6.8: No Outliers Classification Accuracy (70% / 30% manual partitioning based on $|E|$)

	KNN-C	IKNN-C	C_SVM-C	NU_SVM-C	BioHEL-C
Training Set	63.58%	23.18%	80.13%	88.08%	97.39%
Testing Set	70.42%	33.80%	67.61%	66.20%	66.05%

P-value is 3.57×10^{-215} for the training set and 4.53×10^{-178} for the testing set.

When the selection of networks for the training and testing sets is based on $|E|$, the training and testing accuracies are instead those presented in Table 6.8. Although higher than for the $|V|$ based experiments, the testing accuracies do not exceed results observed before. At least with a 70%/30% split, neither $|V|$ nor $|E|$ seem to allow for a distribution of instances that helps in obtaining better classification accuracy.

In figure 6.5, the testing accuracy distribution when running the 250 experiments with BioHEL is shown. The standard deviation observed is 3.48%, with the best result producing 74.65% testing accuracy. In the same way that was observed for the average accuracies of all algorithms, the values in this experiment are near those observed when producing the training and testing sets using Keel automatic 10-fold feature.

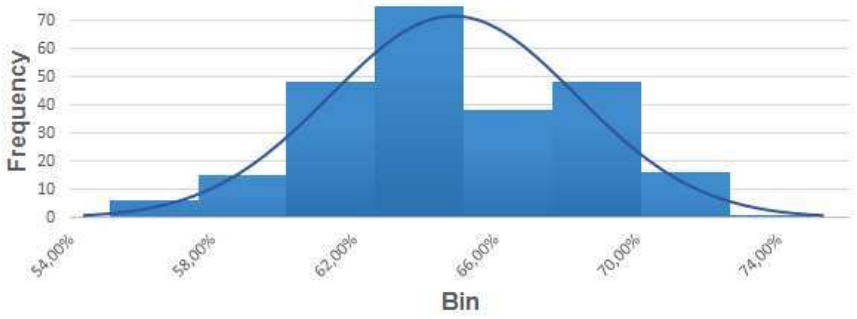


Figure 6.5: Histogram for BioHEL-C testing accuracies when discarding outliers, using 70% / 30% manual partitioning based on $|E|$

Table 6.9: Number of rules and number of attributes per rule, produced by BioHEL when discarding outliers, using 70% / 30% manual partitioning based on $|E|$

Rules per rule set		Attributes per rule	
Average	St. Dev.	Average	St. Dev.
17.58	1.25	4.83	0.96

Table 6.9 shows that the size of the rule sets produced by BioHEL is similar to those observed before.

We proceed now to the following set of experiments, where a 90% of the non-outlier instances are considered for the training set and the remaining 10% are grouped to serve as testing set.

Table 6.10: No Outliers Classification Accuracy (90% / 10% manual partitioning based on $|V|$)

	KNN-C	IKNN-C	C_SVM-C	NU_SVM-C	BioHEL-C
Training Set	67.51%	26.40%	78.17%	80.20%	95.81%
Testing Set	60.00%	36.00%	60.00%	72.00%	58.30%

P-value is 3.57×10^{-215} for the training set and 1.45×10^{-174} for the testing set.

Table 6.10 presents the accuracies when using the partition based on sizes determined by $|V|$. With the exception of the *NU_SVM-C* classification algorithm, the results obtained are comparable with those obtained for the 70%/30% manual partitioning based on $|V|$. The interesting result that differs is the testing accuracy of *NU_SVM-C*, which is the higher obtained with this technique in all the experiments presented so far when using real

networks. All of the values observed are still below the promising accuracies presented for Artificial networks in the Section 5.4 of the previous chapter. Both Support Vector Machines and the BioHEL algorithm continue to provide high training accuracies which are not sustained for the testing sets, hinting at an over-specialization on the features presented by the training instances.

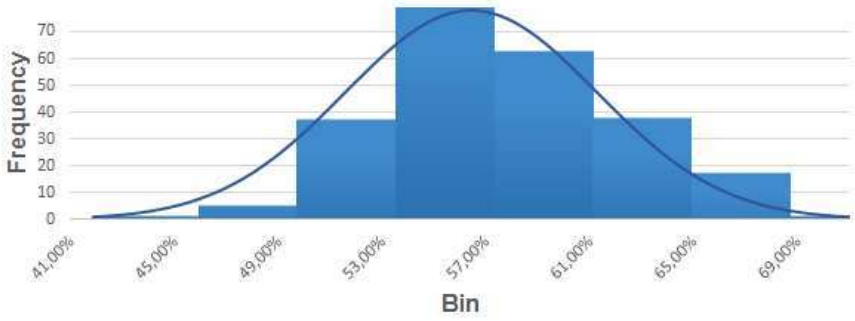


Figure 6.6: Histogram for BioHEL-C testing accuracies when discarding outliers, using 90% / 10% manual partitioning based on $|V|$

Figure 6.6 is the histogram of the testing accuracies presented by BioHEL. The standard deviation recorded is 4.86%, similar to those previously observed. Although the average is not an improvement over previous experiments, the maximum testing accuracy in this experiment when using BioHEL was 72.00%, which means that there are runs when *BioHEL* reached the same results than *NU_SVM-C*. When considering these findings, we can consider that the higher number of samples in the training set can potentially provide a better classification result.

Table 6.11: Number of rules and number of attributes per rule, produced by BioHEL when discarding outliers, using 90% / 10% manual partitioning based on $|V|$

Rules per rule set		Attributes per rule	
Average	St. Dev.	Average	St. Dev.
17.61	1.73	4.17	0.83

When observing Table 6.11, it can be seen that the complexity of the rule sets produced by BioHEL does not change significantly with respect to the experiments previously presented in this section.

When dividing the instances into training and testing sets according to

Table 6.12: No Outliers Classification Accuracy (90% / 10% manual partitioning based on $|E|$)

	KNN-C	IKNN-C	C_SVM-C	NU_SVM-C	BioHEL-C
Training Set	63.96%	19.80%	77.66%	82.74%	95.88%
Testing Set	72.00%	44.00%	60.00%	80.00%	63.39%

P-value is 3.57×10^{-215} for the training set and 2.98×10^{-200} for the testing set.

$|E|$ with a 90%/10% split, the results are those shown in Table 6.12. Testing accuracy for *C_SVM-C* and *BioHEL-C* are slightly below the experiment with 70%/30% split. But the most interesting result is the testing accuracies of the other algorithms. *KNN-C*, *IKNN-C* and *NU_SVM-C* present the highest values for each algorithm across all the experiments performed with real networks. Although the accuracy of *IKNN-C* is still below any result of interest, the most important finding is the testing accuracy of *NU_SVM-C*, providing a strong classification at 80.00%, much closer to the results observed for the artificial networks in the previous chapter.

Considering the last four experiments, it is without doubt that the number of edges provide a more accurate way of partitioning instances when creating training and testing sets for our classification algorithms.

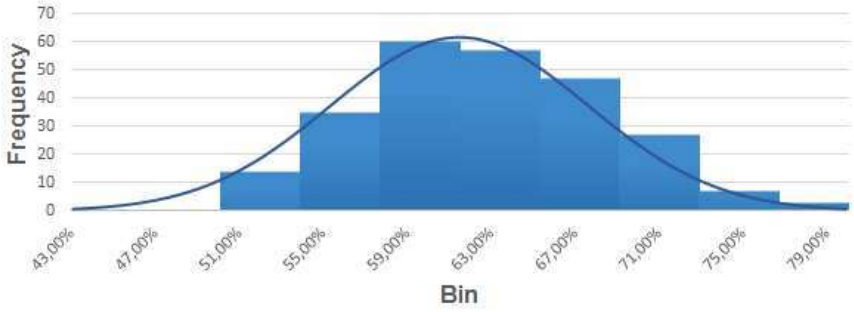


Figure 6.7: Histogram for BioHEL-C testing accuracies when discarding outliers, using 90% / 10% manual partitioning based on $|E|$

The last histogram we need to review is the one presented in Figure 6.7. Although the average testing recorded is 63.39%, the standard deviation in the results was 6.14%. This is in part because the best BioHEL experiment in the 250 runs achieved a testing accuracy of 80.00%, reaching the same result that *NU_SVM-C* for this experiment. This is, to our understanding, an even better outcome, since BioHEL provides us with a set of rules that

allow the classification of the networks in a human readable way, which eases the efforts to apply it to further projects. This best rule set is presented in Table 6.13.

Table 6.13: Best ruleset produced by BioHEL when discarding outliers, using 90% / 10% manual partitioning based on $|E|$

```
00: Betweenness(Q3) is > 0.009807494
    OR Clustering(Max) is < 0.49995282
    THEN Metabolic

01: |E| is > 9.6093665E-4
    OR Closeness(Avg) is > 0.0031226687
    OR Closeness(Q3) is < 0.1692386
    OR Eigenvector(Avg) is < 0.891398
    THEN SpeciesNetworkGraph

02: OffdiagonalComplexity is > 0.10147023
    OR Clustering(Max) is < 0.064023055
    OR Eigenvector(Q3) is > 0.3981593
    THEN CombinatorialDNALibrarySynthesis

03: Clustering(IQR) is > 0.113217935
    OR Clustering(Max) is > 0.44832414
    OR Eigenvector(Min) is > 0.0028145611
    OR PageRank(Q2) is
    BETWEEN (0.055136032 AND 0.98772764)
    THEN Social

04: Closeness(Min) is < 0.9819039
    OR OffdiagonalComplexity is
    BETWEEN (0.14153215 AND 0.43907204)
    OR Clustering(Avg) is
    BETWEEN (0.0010310696 AND 0.63036)
    OR Eigenvector(Q2) is > 0.0177089
    OR Eigenvector(Q3) is > 0.04434352
    OR PageRank(Min) is < 0.30869475
    THEN ReactionDependencyGraph

05: |E| is < 0.4533041
    OR Betweenness(Q2) is < 0.5938384
    OR Betweenness(Min) is < 0.53768516
    OR Closeness(Avg) is
    BETWEEN (0.02464278 AND 0.29275)
    OR Closeness(Q2) is < 0.59215415
    OR Closeness(IQR) is
    BETWEEN (0.27906564 AND 0.6538935)
    OR Closeness(Min) is < 0.74175996
    OR Closeness(Max) is
    BETWEEN (0.16553417 AND 0.62577844)
    OR GlobalEfficiency is < 0.5310019
    OR MaximumDegree is
    BETWEEN (0.050123066 AND 0.6597434)
    OR Clustering(Max) is
    BETWEEN (0.25693303 AND 0.74306697)
    OR Eigenvector(Avg) is < 0.34877324
    OR Eigenvector(Q1) is < 0.6777469
    OR Eigenvector(Q2) is < 0.31767222
    OR Eigenvector(IQR) is < 0.38055673
    OR PageRank(Avg) is < 0.3643762
    OR PageRank(Q2) is < 0.59571415
    OR PageRank(Max) is
    BETWEEN (0.18540487 AND 0.49678728)
    THEN SpeciesNetworkGraph

06: GlobalEfficiency is > 0.005893192
    OR Eigenvector(Q2) is > 0.0048568067
    OR PageRank(Max) is > 0.057587724
    THEN ReactionDependencyGraph

07: Clustering(Q3) is < 0.49987158
    OR PageRank(Q1) is
    BETWEEN (0.052726876 AND 0.5013969)
    THEN Infrastructure

08: OffdiagonalComplexity is < 0.6523598
    OR Clustering(Avg) is > 0.39331004
    OR Clustering(IQR) is > 0.046484824
    OR Eigenvector(Avg) is > 0.07282676
    OR PageRank(IQR) is < 0.84130305
    THEN Tweet

09: |E| is > 0.059126627
    OR Clustering(IQR) is < 0.75337255
    OR Eigenvector(IQR) is > 0.0022295131
    THEN Tweet

10: Closeness(Q1) is > 0.2355396
    OR Clustering(Q3) is > 0.46679175
    THEN Infrastructure

11: Closeness(Max) is > 0.09994781
    OR Clustering(Q3) is > 0.21303262
    OR PageRank(IQR) is < 0.99251384
    OR PageRank(Max) is > 0.13399653
    THEN Collaboration

12: |V| is > 0.0026750874
    OR Closeness(Q2) is > 0.12743585
    OR Closeness(Q3) is
    BETWEEN (0.25736293 AND 0.397376)
    THEN Collaboration

13: Betweenness(Q3) is < 0.014767424
    OR Closeness(Avg) is
    BETWEEN (0.1946039 AND 0.5063392)
    OR Clustering(IQR) is < 0.35929194
    THEN Communication

14: |E| is > 0.015957251
    OR Betweenness(Q3) is < 0.1125778
    THEN Tweet

15: Closeness(Q1) is < 0.38494322
    OR Closeness(Q2) is > 0.0015463152
    OR OffdiagonalComplexity is > 0.3929616
    THEN Code

16: Closeness(Max) is > 0.06881322
    OR PageRank(Avg) is
    BETWEEN (0.0033149263 AND 0.9989146)
    THEN AssortedBiological

17: Closeness(Avg) is
    BETWEEN (0.116268456 AND 0.8539514)
    THEN Infrastructure

18: ELSE Tweet
```

Table 6.14: Number of rules and number of attributes per rule, produced by BioHEL when discarding outliers, using 90% / 10% manual partitioning based on $|E|$

Rules per rule set		Attributes per rule	
Average	St. Dev.	Average	St. Dev.
18.50	1.58	4.87	0.91

Table 6.14 shows, as in previous results, that the number of rules per rule set, and the number of attributes per rule in the BioHEL results is on the same scale across all experiments involving real networks, hinting at an intrinsic complexity of the classification problem, as predicted by the box plot figures presented in Section 4.3.3.

6.6 Outliers as a testing set

To complete the set of experiments, we will once again consider the set of instances previously marked as outliers, using them as the testing set of the experiments presented so far. The idea behind this proposal is to evaluate whether any of the algorithms has been able to build a classification scheme that is applicable as well to the outlier networks for the different domains.

It is our intention to determine if any of the experimental set-ups proposed is conductive to obtaining a resulting classification that is general enough to accommodate for unexpected values, keeping as much as possible its previous accuracy.

Table 6.15: Classification Accuracy for Outliers as testing set

	KNN-C	IKNN-C	C_SVM-C	NU_SVM-C	BioHEL-C
All non-outliers	33.33%	27.78%	16.67%	27.78%	30.02%
70% $\|V\|$	33.33%	22.22%	11.11%	27.78%	29.80%
70% $\|E\|$	38.89%	22.22%	16.67%	16.67%	27.20%
90% $\|V\|$	38.89%	27.78%	16.67%	27.78%	29.71%
90% $\|E\|$	27.78%	16.67%	11.11%	22.22%	26.78%

P-values is 1.70×10^{-154} for *All non-outliers*, 1.26×10^{-185} for 70% $|V|$, 2.98×10^{-192} for 70% $|E|$, 1.31×10^{-180} for 90% $|V|$ and 1.19×10^{-182} for 90% $|E|$.

Table 6.15 presents the summarized testing accuracy for the different experiments when tested against the outlier instances. In it, the algorithms show the expected low accuracy across all experiments. The highest results

are obtained when considering all non-outlier instances, and even in this scenario only *KNN-C* and *BioHEL-C* go above 30% testing accuracy. The maximum accuracy for BioHEL is observed at 50.00%, which is still far below the promising results produced in previous experiments.

With this results it is clear that the techniques proposed still lack the ability to handle unseen instances that are outside the range of the values presented during training.

6.7 Review

In this chapter we finally present and run the set of experiments to check whether it is possible to classify networks, once reduced to a set of complexity measurements, according to their domain of origin.

We proposed a set of experimental schemes which began with simple partitioning in folds using the Keel framework automatic partitioning feature. Outliers in the original data were detected and considered separately in following experiments. Then we detailed a division of the network instances according to their size, as understood from the number of vertices and edges. This division was done twice: considering 70% for the training set and 30% for testing, and then 90% for training with only 10% for testing. Finally we analyzed the possibility of using the same training sets but checking testing accuracy of the classification algorithms against the set of outliers previously distinguished.

We have found that most of the times a simple K Nearest Neighbor classification algorithm obtained the best accuracy, while a more complex technique such as Support Vector Machine or Evolutionary Rule Learning seemed to over-specify on the characteristics of the training instances. The latter provide an almost perfect training accuracy, but testing accuracy results that were on par and sometimes below the values observed for the simpler K Nearest Neighbor classifier.

Our 70/30 splitting technique for training and testing instances did not improve the results obtained in the experiment with automatically generated folds, and gave even lower accuracy when vertices were used as the set building criteria. When splitting 90/10 considering number of edges we were able to obtain testing accuracy results that reached values nearer the 80% observed for the artificial networks in Chapter 5. Throughout all the experiments we were able to confirm that BioHEL, the proposed Evolutionary Rule Learning algorithm, offered a low deviation in accuracy over the different runs performed, and managed to offer a maximum result that

was always near the top of the observed accuracies for all algorithms. This, coupled with the fact that it provides as a result a set of clear rules for obtaining the accuracy reported, allows us to propose it as an effective tool in scenarios such as the one reviewed in this work.

Capítulo 6: Experimentos

Este capítulo define un conjunto de experimentos para analizar la factibilidad de resolver el problema de clasificación propuesto. Luego de cada experimento se presentan los resultados obtenidos, y un análisis de lo observado.

El primer objetivo es recrear los experimentos del capítulo 5, pero en este caso utilizando las redes reales presentadas en la sección 4.2.

Para analizar nuestro problema de clasificación, cada red es representada por una colección de valores numéricos entre 0 y 1 (usando la normalización explicada en la sección 4.3.3), y un atributo adicional que indica el dominio al que pertenece. Denominamos a cada una de estas representaciones "instancia" del problema de clasificación. Luego estas instancias son distribuidas con distintos criterios en conjuntos de entrenamiento y de prueba.

Adicionalmente, en la sección 6.3 proponemos una técnica para la detección de *outliers*.

El objetivo final es la confirmación de nuestra hipótesis: dado un conjunto de redes de distintos dominios, es posible encontrar cualidades que definen cada dominio, a través del estudio de métricas de complejidad calculadas a partir de las redes.

Los algoritmos aplicados son los mismos que se mencionaron previamente en la tabla 5.2, utilizando los parámetros por defecto, los cuales se resumen en la tabla 5.3. Presentamos resultados que representan el promedio de 250 ejecuciones, y realizamos un análisis estadístico para garantizar la significancia de los resultados.

Los primeros experimentos, presentados en la sección 6.2, utilizan la partición automática de Keel para generar conjuntos de entrenamiento y prueba.

Luego, en la sección 6.3 se determinan los *outliers* en el conjunto original de redes para considerarlos por separado.

A continuación proponemos un criterio manual de separación de instancias, primero en base a cantidad de vértices y luego en base a la cantidad de ejes. Con estos criterios, construimos un esquema donde el 70% de las redes son utilizadas para entrenamiento y el 30% restante para prueba, así como también usando 90% y 10% respectivamente.

Finalmente presentamos experimentos donde los *outliers* son utilizados como conjunto de prueba, para estudiar la capacidad de generalización de las clasificaciones obtenidas.

En general, observamos que el algoritmo de clasificación **K Nearest Neighbor** (el más simple de los propuestos) obtiene la mayor precisión, ya que los otros algoritmos parecen sobre-especificar su aprendizaje para las características del conjunto de entrenamiento.

Los resultados más interesantes se obtienen para el caso en que particionamos las instancias en 90/10 considerando la cantidad de ejes (tabla 6.12). En el mismo la precisión sobre el conjunto de prueba se acerca al 80% como habíamos observado para las redes artificiales en el capítulo 5.

A lo largo de todos los experimentos pudimos observar que el sistema **BioHEL** tiende a ofrecer entre sus ejecuciones algunos resultados que están entre las precisiones más altas, con el beneficio adicional de que su salida está conformada por reglas que son de fácil lectura, ayudando a la comprensión de lo estudiado.

Chapter 7

Conclusions and Discussions

7.1 Description

In this final chapter we will review the contributions of the work presented, and summarize the different conclusions observed in previous chapters.

With that we will present an analysis of the feasibility of our original proposal, along with the relevant observations that can be gathered from the findings of the different experiments presented.

Afterwards we conclude with a number of possible topics for future work.

7.2 Contributions

Before reviewing all the findings in each chapter, we mention first the most relevant contributions that this work has produced.

- Exhaustive evaluation of algorithms to calculate network properties.
- Extensive analysis of properties and their correlations within networks domains and across domains.
- Demonstration that simple classification algorithms can utilize some of these network properties to classify a large number of artificial networks.
- Demonstration that this can be taken further to classify real networks.

7.3 Findings

We started this work with the intention of determining whether there is an underlying equivalence in different networks from a given domain, such that they can be observed, learned, and used then in classifying it. In a way, we want to understand if there is an "identity" to the graph that represent a specific subject in the form of a network.

To achieve that, we started by proposing a large number of complexity measures from the literature. Since we need a way to compare networks of completely different origins, graph shapes and characteristics, we determined that they would be require to minimize the input data for possible experiments. In Section 2.4.4, we presented the summary of the formulas reviewed, grouped by considering their focus when analyzing the graph structure.

We presented a set of 3 different approaches to classification problems: *K Nearest Neighbors*, *Support Vector Machines* and *Evolutionary Rule Learning*. These algorithms were found to be available in the *Keel* framework, which has allowed us to propose comparable experiments that apply them.

With the goal of classifying networks in different domains, we reviewed a number of scientific papers, journals and open network repositories to create a well balanced data set, with 11 domains and a total of 240 networks. The visualization of some sample graphs from those selected allowed a more clear understanding of the complex structures that are involved in our work. These figures also helped us learn how complexity increases in the different domains as the number of vertices and edges grow in number.

Some of the formulas studied required specific information only available for graphs obtained from specific domains, or expensive computations that defeated the idea of simplifying the analysis of a large number of network instances. With that in mind, we obtained in Section 4.3 a list of 13 selected measures that were deemed suitable for our experiments. Since some of them offer values for each vertex in a network, they were further converted into derived measures for average, maximum, minimum, and quartile distribution values, giving a total of 43 graph measures, which were then computed to each of our 240 networks. We were able to confirm their correlation by domain and in general, and found that although a considerable correlation existed between some of them in specific domains, the overall values were smoothed out. We presented then a statistical distribution of the recorded measures, which allowed a preliminary analysis of the possibilities that classification techniques would have if attempting to distinguish

the domains chosen when considering the networks provided.

With the use of the artificial network generator *CiGRAM*, 20 random sample networks were built for each of 8 different classes, in a way to create a synthetic benchmark for our experiments. The statistical distribution of these networks by class was recorded and presented. With the resulting charts we were able to find that of the 8 artificial network classes, two pairs of them displayed overlapping values, which was mentioned as a possible source of confusion for the classification algorithms. In reviewing the experimental results in Section 5.4, we found this hypothesis proven true, as the most significant reduction in testing accuracy was observed for the 4 highlighted classes. At this time we could conclude that the algorithms were having trouble in differentiating networks with different number of communities when their density was higher. Despite this observation, an overall accuracy of 80% was achieved for most of the algorithms. Moreover, we could obtain the set of rules created by *BioHEL*, the Evolutionary Rule Learning in use, and observe that it explicitly employed the larger part of its best solution to differentiate between the 4 classes that presented the overlap.

Once we had confirmed the feasibility of our proposed experiments for classifying artificial networks according to their designated class, we focused on running equivalent scenarios, this time considering the real world networks reviewed in Section 4.2.

A different set of experiments were proposed and their results analyzed. First all network instances were divided in 10 folds and then further into training and testing sets using Keel automatic fold building feature, which does not apply any specific criteria to the distribution. Except for one algorithm (IKNN-C), which produces low quality results for all of the experiments in Chapter 6, we obtained testing accuracies that hardly surpassed 60%, considerably below the values observed previously for artificial networks. We proposed retrying the automatic fold experiment, this time excluding network instances that were deemed as outliers, with no improvement in the results.

In order to analyze the relevance of supplying the classification algorithms with an even distribution of samples when training, we proposed additional experiments which specified a 70% of the networks in each class to be considered for the training set, evenly selected according to number of vertices and also another experiment using the number of edges instead. We observed slightly increased testing accuracies that got nearer to 70%, but only when we selected the networks considering number of edges.

A better result was finally observed when 90% of the instances were used

for the training set, specially when the distribution was made according to the number of edges. In this opportunity, one of the Support Vector Machine algorithms (*NU_SVM-C*) obtained an 80% testing accuracy, a value comparable to the one displayed during the classification of artificial networks. BioHEL also produced for this experiment results that achieved 80% testing accuracy in one of the 250 runs that were executed. This fact implied that we could present a set of human-readable rules that allow this classification accuracy when applied to the network instances considered, without the need to re-run any experiment, since they can be easily applied by translating them into a specific algorithm.

We also attempted to classify the network instances determined to be outliers from our original set, and found that no one of the classification algorithms employed was able to surpass a 50% accuracy when classifying them. This implied that we should discard for now the possibility of considering that the classification ability of the algorithms is able to understand instances that are far from the representative values in terms of complexity measures.

We can conclude that our objective of attempting to classify networks according to their domains of origin is a concrete possibility, with the added value that we could reproduce similar results for artificial and real networks. Furthermore, the observed statistical distribution of the complexity measures provided a consistent preview of the classification possibilities for the domains considered, both in artificial and in real world domains.

7.4 Future Work

With the ideas, experimental proposals and results obtained in this work, there remain a number of interesting directions in which to continue the research of using network complexity measures to classify their problem domain.

1. *Alternative Categories*: an interesting subject of study would be to understand if the domain problems chosen for this work allow for a complete understanding of the possible results that could be obtained with the techniques proposed. Since some of the categories were more troublesome for the classification algorithm than others, it would be interesting to determine a criteria that allows a given domain problem to be labeled as "feasible for classification through complexity measures" or not.

2. *Sub-Domain Categories*: this work sets the ground for analyzing the classification possibilities of some domain problems. We have proposed in the beginning of this work that the same ideas could be applied to more specific networks, using sub-domain problems as categories.

3. *Different Selection of Measures*: we have reviewed a large number of measures. Only some of them were chosen, and then expanded with statistical alternatives such as average, inter-quartile range, minimum, etc. A next step in this direction would be to try different combinations of complexity measures, and perform feature selection by iteratively choosing the ones that are more conductive to higher accuracies. The compact rule results provided by BioHEL could help choose only those that are referenced in the resulting rulesets.

4. *Better artificial networks*: reading the resulting rulesets built by BioHEL, we have at our disposal a range of expected values for the measures that were relevant for the classification task. Some of these measures could be considered in artificial network providers such as the CiGRAM framework used in this work. This means that researchers could create as many artificial networks as required that present properties known to exist in real networks for their domain problem.

5. *Focus on branch decomposition techniques*: Measures based on the concept of branch decomposition[162, 78, 153] (such as Tree-width, Branch-width, Path-width, Carving-width, Clique-width, etc.), have not been reviewed in this thesis, but offer a whole range of possibilities. Although their optimal solution is an NP-hard optimization problem, different solvers exist that could be considered and explored using some of the ideas studied in the previous chapters.

6. *Alternative result analysis*: Apart from the accuracy computed for the different experiments, different techniques exist that would focus on other aspects of the ideas proposed. For instance, the use of measures such as Fleiss' Kappa[77, 93] or Matthews Correlation Coefficient[130] would give an alternative interpretation to the results of the different solvers and determine their relevance.

7. *Alternative correlation analysis*: Instead of computing the correlation of the different measures using Pearson correlation[157], alternative formulas such as Spearman[141] could be applied and the differences studied.

Capítulo 7: Conclusiones y propuestas

El capítulo final de esta tesis repasa las contribuciones del trabajo realizado, presenta un resumen de las observaciones obtenidas en capítulos previos, y enumera propuestas de trabajo futuro.

Contribuciones

- Evaluación exhaustiva de algoritmos para cálculo de propiedades de redes.
- Extenso análisis de propiedades y su correlación para redes de un mismo dominio, así como entre dominios.
- Demostración de que algoritmos de clasificación sencillos son capaces de utilizar las propiedades de las redes para clasificar un gran número de redes artificiales.
- Demostración de que es posible extender la clasificación a redes reales.

Hallazgos

Comenzamos este trabajo con la intención de determinar si era posible encontrar una equivalencia entre redes que pertenecen a un mismo dominio, de manera de observarlas y usarlas para su posterior clasificación. Para ello se propuso estudiar un gran número de métricas de complejidad de la literatura. Las mismas fueron presentadas y agrupadas de acuerdo al enfoque que implican al analizar la estructura de un grafo.

Estudiamos 3 alternativas para resolver problemas de clasificación: *K Nearest Neighbors*, *Support Vector Machines* y *Evolutionary Rule Learning*. Presentamos a su vez el ambiente de experimentación *Keel*, el cual nos permitió ofrecer resultados comparables tanto en la construcción de los experimentos como en la salida de las ejecuciones.

Para estudiar la capacidad de estos algoritmos a la hora de clasificar, obtuvimos de distintas fuentes un total de 240 redes, las cuales consideramos como pertenecientes a uno de 11 dominios distintos. Se presentó una visualización de algunas redes en cada dominio para ayudar a la comprensión de su estructura.

De las métricas relevadas, elegimos un subconjunto de 13 que presentaban facilidad de cómputo y baja correlación entre ellas. Aquellas que trabajan sobre nodos fueron derivadas en base a su máximo, mínimo, promedio y cuartiles, para darnos un total de 43 métricas sobre grafos. Se realizó un análisis estadístico completo sobre las 240 redes y se observó la distribución de los valores obtenidos.

A través del generador de redes artificiales *CiGRAM* se creó un conjunto de 160 redes artificiales pertenecientes a 8 dominios distintos, con el objetivo de estudiar una distribución similar a la ofrecida por las redes reales relevadas. Las mismas fueron sometidas al mismo análisis estadístico y de correlación. Pudimos observar que 2 pares de dominios presentaban valores solapados en todas las métricas elegidas, lo cual fue postulado como un posible factor que complicaría la tarea de ser distinguidas por los algoritmos de clasificación presentados. Esta hipótesis resultó cierta, ya que si bien los algoritmos lograron en general un rendimiento en su precisión del 80%, los mayores problemas en la discriminación fueron observados en los dominios que presentaban valores solapados. Adicionalmente pudimos observar las ventajas de trabajar con el sistema de aprendizaje por reglas *BioHEL*, el cual ofreció un conjunto de reglas legibles donde pudo apreciarse la mayor dificultad para distinguir entre los dominios artificiales con valores solapados.

Al momento de analizar la factibilidad de clasificar las redes reales, propusimos distintos experimentos. El primero implicó utilizar el método de particionamiento automático de Keel, con el cual apenas se sobrepasó el 60% de rendimiento. Resultados similares se obtuvieron luego de eliminar las redes que fueron identificadas como *outliers*.

Para estudiar la relevancia de la distribución de las instancias entre los conjuntos de entrenamiento y prueba, propusimos una asignación manual, donde el 70% fue utilizada en el primero y el 30% restante en el segundo. Para esta asignación se usaron dos criterios: de acuerdo a la cantidad de vértices en cada grafo, y luego en base a la cantidad de ejes. Considerando la cantidad de ejes, se logró mejorar en estos experimentos el rendimiento a valores más cercanos al 70%.

Repetimos la idea de construir manualmente los conjuntos, esta vez asignando 90% al conjunto de entrenamiento y 10% al conjunto de prueba. En este caso, al aplicar el criterio de cantidad de ejes, una de las variantes de Support Vector Machine (*NU_SVM-C*) logró una precisión del 80%, similar a lo obtenido para las redes artificiales. BioHEL también obtuvo este rendimiento en una de las ejecuciones para este experimento. A través de

la lectura del conjunto de reglas legibles que generó, pudimos observar la factibilidad de aplicar las mismas en experimentos futuros, sin necesidad de volver a ejecutar los algoritmos de clasificación.

En un último experimento, propusimos reusar los conjuntos de entrenamiento previamente analizados, y un conjunto de pruebas que esté compuesto únicamente por los grafos identificados como *outliers*. En este caso, menos del 50% fueron clasificados correctamente, lo cual implica que los esquemas sugeridos hasta el momento no permiten manejar este tipo de situaciones.

Concluimos que nuestro objetivo de clasificar redes de acuerdo a su dominio de origen utilizando métricas de complejidad es una posibilidad concreta, con el valor agregado de haber podido observar resultados similares tanto en redes artificiales como reales. Adicionalmente, las distribuciones estadísticas de las métricas de complejidad generadas nos permitieron obtener una predicción consistente del potencial de clasificar los dominios.

Trabajo futuro

Las ideas presentadas en este trabajo y los resultados obtenidos, nos permiten proponer las siguientes ideas para encarar trabajos futuros en la tarea de clasificar redes utilizando métricas de complejidad.

1. *Categorías alternativas*: expandir el análisis realizado a otros dominios. En base a lo observado en el presente trabajo, será de gran interés comprender si existen dominios más o menos factibles de ser categorizados a través de métricas de complejidad.

2. *Sub-dominios como categorías*: hemos propuesto en el comienzo de este trabajo la idea de aplicar técnicas similares a las presentadas para distinguir redes que pertenecen a distintos sub-dominios.

3. *Métricas alternativas*: esta tesis presenta un análisis sobre un gran número de métricas, de las cuales sólo algunas fueron elegidas para el proceso de clasificación. Uno de los siguientes pasos sería estudiar otras de las fórmulas relevadas. Utilizando los resultados de los conjuntos de reglas provisto por BioHEL, sería factible plantear un esquema iterativo donde se identifiquen las de mayor efectividad en la

clasificación.

4. *Mejores redes artificiales*: Con los resultados provistos por BioHEL, tenemos a nuestra disposición el detalle de las métricas que fueron de mayor relevancia en la clasificación. Alguna de ellas son en particular parámetros disponibles en la construccíon de redes artificiales del sistema CiGRAM utilizado. Esto significa que un investigador podría crear tantas redes artificiales como sean necesarias y garantizar que las propiedades de las mismas la identifiquen con el dominio real al que simulan pertenecer.

5. *Aplicar Técnicas de descomposición de ramas*: La mediciones basadas en el concepto de decomposición de ramas no fueron estudiadas en esta tesis, pero ofrecen una gran cantidad de posibilidades. Aunque su solución óptima es un problema NP-hard, existen soluciones exactas que podrían considerarse usando las ideas estudiadas en los capítulos anteriores.

6. *Análisis alternativo de resultados*: además de la precisión calculada en los distintos experimentos, existen otras técnicas que permitirían presentar otros aspectos de las ideas presentadas. Por ejemplo, mediciones como Fleiss' Kappa[77, 93] o el coeficiente de correlación de Matthews[130] permitirían una interpretación alternativa de los resultados de los distintos clasificadores.

7. *Análisis alternativo de correlaciones*: En lugar de calcular la correlación entre los valores de las mediciones usando la fórmula de Pearson[157], se podrían aplicar fórmulas alternativas como la de Spearman[141], y estudian las diferencias con respecto a lo obtenido.

Appendix A

Appendices

A.1 Manual partitioning

This is the list of instances that were chosen for each of the manual partitioning experiments in Section 6.5.

When sorting based on the number of vertices, the instances shown in Table A.1 are those that were selected to represent the 30% of the non-outlier inputs to be used as the testing set. Table A.2 presents the selected 10% for the equivalent experiment.

For the sorting based on the number of edges, the list are presented in Tables A.3 and A.4 for 30% and 10% respectively.

The training sets for each experiment were built by removing the instances mentioned in the previous table from the complete list of non-outliers.

Table A.1: Manually selected 30% based on $|V|$

Network	Domain
example-Anew-1-2_7	CombinatorialDNALibrarySynthesis
ukb-Anew-1-5_110	CombinatorialDNALibrarySynthesis
unott_azurin-Anew-1-3_170	CombinatorialDNALibrarySynthesis
uh1-A1-1-3_304	CombinatorialDNALibrarySynthesis
uh2-A1-1-6_515	CombinatorialDNALibrarySynthesis
iGE2008-Anew-1-5_836	CombinatorialDNALibrarySynthesis
C Elegans	AssortedBiological
NT-bio-diseasome	AssortedBiological
NT-bio-yeast-protein-inter	AssortedBiological
Github PHP	Code
Github Python	Code
Github Europe	Code
Political Books	Collaboration
CPAN Authors	Collaboration
CPAN Distributions	Collaboration
infect-hyper	Communication
infect-dublin	Communication
fb-messages	Communication
wiki-trust-dir	Communication
Top-Australia	Infrastructure
Top-Canada	Infrastructure
Top-Chile	Infrastructure
Top-Armenia	Infrastructure
USAir97	Infrastructure
Power Grid	Infrastructure
C. trachomatis	Metabolic
T. pallidum	Metabolic
A. thaliana	Metabolic
T. maritima	Metabolic
C. jejuni	Metabolic
A. actinomycetem comitans	Metabolic
M. bovis	Metabolic
N. gonorrhoeae	Metabolic
M. jannaschii	Metabolic
A. fulgidus	Metabolic
Synechocystis sp.	Metabolic
M. tuberculosis	Metabolic
B. subtilis	Metabolic
BIOMD0000000139-RDG	ReactionDependencyGraph
BIOMD0000000147-RDG	ReactionDependencyGraph
BIOMD0000000268-RDG	ReactionDependencyGraph
BIOMD0000000334-RDG	ReactionDependencyGraph
BIOMD0000000344-RDG	ReactionDependencyGraph
BIOMD0000000175-RDG	ReactionDependencyGraph
BIOMD0000000338-RDG	ReactionDependencyGraph
BIOMD0000000152-RDG	ReactionDependencyGraph
BIOMD0000000049-RDG	ReactionDependencyGraph
Karate Club	Social
School 1	Social
EuroSiS	Social
advogato	Social
gplus	Social
sign-Slashdot090216	Social
BIOMD0000000103-SNG	SpeciesNetworkGraph
BIOMD0000000256-SNG	SpeciesNetworkGraph
BIOMD0000000303-SNG	SpeciesNetworkGraph
BIOMD0000000333-SNG	SpeciesNetworkGraph
BIOMD0000000056-SNG	SpeciesNetworkGraph
BIOMD0000000339-SNG	SpeciesNetworkGraph
BIOMD0000000326-SNG	SpeciesNetworkGraph
BIOMD0000000332-SNG	SpeciesNetworkGraph
BIOMD0000000049-SNG	SpeciesNetworkGraph
NT-rt_assad	Tweet
NT-rt_obama	Tweet
NT-rt_tlot	Tweet
NT-rt_alwefaq	Tweet
NT-rt_bahrain	Tweet
NT-rt_oman	Tweet
NT-rt_dash	Tweet
NT-rt_mittromney	Tweet
NT-rt_http	Tweet

Table A.2: Manually selected 10% based on $|V|$

Network	Domain
ukb-Anew-1-5_110	CombinatorialDNALibrarySynthesis
uh2-A1-1-6_515	CombinatorialDNALibrarySynthesis
NT-bio-diseasome	AssortedBiological
Github Python	Code
CPAN Authors	Collaboration
radoslaw-email	Communication
reality	Communication
Top-Canada	Infrastructure
USAir97	Infrastructure
C. trachomatis	Metabolic
T. maritima	Metabolic
M. bovis	Metabolic
A. fulgidus	Metabolic
B. subtilis	Metabolic
BIOMD0000000303-RDG	ReactionDependencyGraph
BIOMD0000000344-RDG	ReactionDependencyGraph
BIOMD0000000019-RDG	ReactionDependencyGraph
Jazz	Social
gplus	Social
BIOMD0000000256-SNG	SpeciesNetworkGraph
BIOMD0000000056-SNG	SpeciesNetworkGraph
BIOMD0000000332-SNG	SpeciesNetworkGraph
NT-rt_damascus	Tweet
NT-rt_bahrain	Tweet
NT-rt_onedirection	Tweet

Table A.3: Manually selected 30% based on $|E|$

Network	Domain
example-Anew-1-2_7	CombinatorialDNALibrarySynthesis
ukb-Anew-1-5_110	CombinatorialDNALibrarySynthesis
unott_azurin-Anew-1-3_170	CombinatorialDNALibrarySynthesis
uh1-A1-1-3_304	CombinatorialDNALibrarySynthesis
uh2-A1-1-6_515	CombinatorialDNALibrarySynthesis
iGE2008-Anew-1-5_836	CombinatorialDNALibrarySynthesis
NT-bio-diseasome	AssortedBiological
NT-bio-celegans-dir	AssortedBiological
NT-bio-yeast-protein-inter	AssortedBiological
Photoviz	Code
Github Python	Code
Github Europe	Code
Political Books	Collaboration
NT-ca-CSphd	Collaboration
NT-ca-Erdos992	Collaboration
enron-only	Communication
email-univ	Communication
reality	Communication
email-EU	Communication
Top-Canada	Infrastructure
Top-Argelia	Infrastructure
Top-Chile	Infrastructure
Top-Italy	Infrastructure
USAir97	Infrastructure
openflights	Infrastructure
C. trachomatis	Metabolic
T. pallidum	Metabolic
A. thaliana	Metabolic
T. maritima	Metabolic
C. jejuni	Metabolic
A. actinomycetem comitans	Metabolic
M. bovis	Metabolic
N. gonorrhoeae	Metabolic
S. pneumoniae	Metabolic
A. fulgidus	Metabolic
H. influenzae	Metabolic
S. cerevisiae	Metabolic
B. subtilis	Metabolic
BIOMD0000000139-RDG	ReactionDependencyGraph
BIOMD0000000303-RDG	ReactionDependencyGraph
BIOMD0000000147-RDG	ReactionDependencyGraph
BIOMD0000000338-RDG	ReactionDependencyGraph
BIOMD0000000256-RDG	ReactionDependencyGraph
BIOMD0000000334-RDG	ReactionDependencyGraph
BIOMD0000000223-RDG	ReactionDependencyGraph
BIOMD0000000175-RDG	ReactionDependencyGraph
BIOMD0000000109-RDG	ReactionDependencyGraph
firm-hi-tech	Social
wiki-Vote	Social
EuroSiS	Social
gplus	Social
sign-Slashdot090216	Social
wiki-elec	Social
BIOMD0000000139-SNG	SpeciesNetworkGraph
BIOMD0000000147-SNG	SpeciesNetworkGraph
BIOMD0000000339-SNG	SpeciesNetworkGraph
BIOMD0000000250-SNG	SpeciesNetworkGraph
BIOMD0000000230-SNG	SpeciesNetworkGraph
BIOMD0000000334-SNG	SpeciesNetworkGraph
BIOMD0000000223-SNG	SpeciesNetworkGraph
BIOMD0000000152-SNG	SpeciesNetworkGraph
BIOMD0000000153-SNG	SpeciesNetworkGraph
NT-rt_voteonedirection	Tweet
NT-rt_damascus	Tweet
NT-rt_israel	Tweet
NT-rt_islam	Tweet
NT-rt_libya	Tweet
NT-rt_uae	Tweet
NT-rt_bahrain	Tweet
NT-rt_onedirection	Tweet
NT-rt_justinbieber	Tweet

Table A.4: Manually selected 10% based on $|E|$

Network	Domain
ukb-Anew-1-5_110	CombinatorialDNALibrarySynthesis
uh2-A1-1-6_515	CombinatorialDNALibrarySynthesis
NT-bio-celegans-dir	AssortedBiological
Github Python	Code
NT-ca-CSphd	Collaboration
infect-dublin	Communication
Mails	Communication
Top-Argelia	Infrastructure
USAir97	Infrastructure
C. trachomatis	Metabolic
T. maritima	Metabolic
M. bovis	Metabolic
A. fulgidus	Metabolic
B. subtilis	Metabolic
BIOMD0000000250-RDG	ReactionDependencyGraph
BIOMD0000000256-RDG	ReactionDependencyGraph
BIOMD0000000152-RDG	ReactionDependencyGraph
Jazz	Social
sign-Slashdot090216	Social
BIOMD0000000147-SNG	SpeciesNetworkGraph
BIOMD0000000230-SNG	SpeciesNetworkGraph
BIOMD0000000152-SNG	SpeciesNetworkGraph
NT-rt_obama	Tweet
NT-rt_libya	Tweet
NT-rt_mittromney	Tweet

A.2 Experimental Training and Testing Accuracies Detail

We present here the confusion matrices for the training and testing sets of the different experiments executed in Chapter 6 when considering the real networks.

The abbreviations used throughout the tables are shown in Table A.5

Table A.5: Abbreviations names for the real network domains in the experimental result tables

Domain	Abbreviation
Assorted Biological	A.Bio.
Code	Code
Collaboration	Collab.
Combinatorial DNA Library Synthesis	C.DNA
Communication	Comm.
Infrastructure	Infra.
Metabolic	Meta.
Reaction Dependency Graph	RDG
Social	Social
Species Network Graph	SNG
Tweet	Tweet

Table A.6: All Instances (Automatic Partitioning) KNN-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	88%	0%	0%	0%	0%	11%	0%	1%	0%	0%	0%
A.Bio.	0%	19%	10%	9%	10%	12%	0%	11%	0%	10%	19%
Code	0%	2%	56%	2%	0%	0%	0%	1%	1%	18%	20%
Collab.	0%	14%	15%	21%	11%	9%	0%	0%	0%	8%	23%
Comm.	0%	2%	0%	10%	13%	20%	0%	7%	13%	1%	34%
Infra.	5%	18%	1%	0%	2%	43%	1%	14%	1%	0%	17%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	4%	0%	0%	3%	0%	11%	0%	78%	0%	3%	0%
Tweet	0%	0%	5%	1%	0%	0%	0%	5%	73%	0%	18%
RDG	3%	6%	3%	0%	0%	0%	0%	0%	0%	87%	0%
SNG	3%	6%	9%	10%	5%	6%	0%	0%	14%	3%	44%

Table A.7: All Instances (Automatic Partitioning) KNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	89%	0%	0%	0%	0%	11%	0%	0%	0%	0%	0%
A.Bio.	0%	20%	10%	20%	10%	10%	0%	10%	0%	0%	20%
Code	0%	0%	60%	0%	0%	0%	0%	0%	0%	20%	20%
Collab.	0%	15%	15%	23%	15%	8%	0%	0%	0%	8%	15%
Comm.	0%	0%	0%	14%	14%	21%	0%	0%	14%	0%	36%
Infra.	5%	14%	0%	5%	0%	45%	0%	14%	0%	0%	18%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	3%	0%	0%	3%	0%	7%	0%	80%	3%	3%	0%
Tweet	0%	0%	5%	0%	0%	0%	0%	5%	74%	0%	16%
RDG	3%	3%	7%	0%	0%	0%	0%	0%	0%	87%	0%
SNG	3%	6%	10%	6%	10%	6%	0%	0%	13%	3%	42%

Table A.8: All Instances (Automatic Partitioning) IKNN-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	32%	0%	0%	0%	0%	17%	0%	30%	0%	21%	0%
A.Bio.	0%	4%	9%	9%	10%	26%	0%	10%	0%	10%	22%
Code	0%	2%	16%	16%	0%	16%	0%	1%	8%	18%	24%
Collab.	0%	14%	9%	8%	15%	17%	0%	0%	0%	2%	36%
Comm.	0%	6%	1%	26%	6%	14%	0%	6%	9%	0%	33%
Infra.	2%	15%	2%	0%	11%	35%	0%	17%	1%	0%	19%
Meta.	0%	16%	0%	0%	0%	0%	19%	59%	0%	0%	6%
Social	1%	3%	0%	4%	4%	23%	0%	49%	7%	3%	5%
Tweet	0%	1%	1%	5%	11%	2%	0%	4%	37%	0%	40%
RDG	1%	7%	3%	19%	7%	0%	0%	0%	0%	31%	32%
SNG	1%	5%	7%	10%	27%	8%	0%	0%	19%	2%	22%

Table A.9: All Instances (Automatic Partitioning) IKNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	33%	0%	0%	0%	0%	11%	0%	39%	0%	17%	0%
A.Bio.	0%	20%	10%	20%	10%	10%	0%	10%	0%	0%	20%
Code	0%	0%	50%	0%	0%	0%	0%	0%	10%	20%	20%
Collab.	0%	23%	15%	23%	8%	8%	0%	0%	0%	0%	23%
Comm.	0%	7%	0%	14%	21%	21%	0%	0%	7%	0%	29%
Infra.	5%	9%	0%	5%	5%	50%	0%	9%	0%	0%	18%
Meta.	0%	9%	0%	0%	0%	0%	23%	58%	0%	0%	9%
Social	3%	0%	0%	3%	0%	17%	0%	67%	7%	3%	0%
Tweet	0%	0%	5%	0%	5%	0%	0%	5%	58%	0%	26%
RDG	0%	3%	7%	20%	10%	0%	0%	0%	0%	23%	37%
SNG	0%	13%	13%	3%	16%	6%	0%	0%	13%	3%	32%

Table A.10: All Instances (Automatic Partitioning) SVM-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	1%	21%	7%	20%	4%	26%	0%	4%	0%	0%	17%
Code	0%	0%	33%	1%	0%	20%	0%	0%	0%	9%	37%
Collab.	0%	1%	9%	49%	0%	17%	0%	8%	0%	8%	9%
Comm.	0%	0%	0%	11%	23%	30%	0%	0%	6%	1%	29%
Infra.	5%	1%	5%	4%	1%	69%	0%	10%	0%	1%	6%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	9%	0%	91%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	96%	0%	4%
RDG	0%	0%	0%	0%	0%	3%	0%	0%	0%	97%	0%
SNG	0%	0%	2%	7%	1%	10%	0%	3%	17%	3%	57%

Table A.11: All Instances (Automatic Partitioning) SVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	10%	0%	10%	20%	0%	40%	0%	10%	0%	0%	10%
Code	0%	0%	20%	10%	0%	20%	0%	0%	0%	10%	40%
Collab.	0%	15%	8%	23%	0%	23%	0%	8%	0%	8%	15%
Comm.	0%	0%	0%	21%	7%	29%	0%	0%	7%	7%	29%
Infra.	9%	5%	5%	9%	5%	41%	0%	14%	0%	0%	14%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	13%	0%	80%	0%	0%	7%
Tweet	0%	0%	0%	0%	0%	0%	0%	5%	79%	0%	16%
RDG	3%	0%	0%	0%	0%	7%	0%	0%	0%	90%	0%
SNG	3%	0%	3%	10%	3%	6%	0%	0%	23%	3%	48%

Table A.12: All Instances (Automatic Partitioning) NUSVM-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	2%	74%	1%	9%	3%	7%	0%	1%	0%	1%	1%
Code	0%	8%	76%	2%	2%	0%	0%	0%	0%	0%	12%
Collab.	0%	21%	21%	32%	7%	6%	0%	1%	0%	5%	8%
Comm.	0%	9%	5%	22%	48%	7%	0%	0%	0%	0%	9%
Infra.	1%	4%	2%	6%	4%	72%	2%	5%	0%	1%	6%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	1%	0%	3%	0%	95%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	96%	0%	4%
RDG	0%	0%	0%	0%	0%	1%	0%	0%	0%	99%	0%
SNG	1%	11%	5%	5%	12%	3%	0%	0%	13%	3%	46%

Table A.13: All Instances (Automatic Partitioning) NUSVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	10%	10%	10%	30%	0%	20%	0%	0%	0%	0%	20%
Code	0%	10%	50%	0%	10%	0%	0%	0%	0%	10%	20%
Collab.	0%	23%	23%	15%	8%	15%	0%	0%	0%	8%	8%
Comm.	0%	7%	0%	21%	29%	14%	0%	0%	14%	7%	7%
Infra.	5%	9%	9%	5%	5%	41%	5%	23%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	3%	0%	0%	0%	10%	3%	80%	0%	3%	0%
Tweet	0%	0%	0%	5%	0%	0%	0%	5%	84%	0%	5%
RDG	3%	3%	3%	3%	3%	0%	0%	0%	0%	83%	0%
SNG	3%	16%	6%	13%	16%	3%	0%	0%	26%	3%	13%

Table A.14: All Instances (Automatic Partitioning) BioHEL-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	61%	1%	21%	7%	1%	0%	0%	0%	0%	10%
Code	0%	0%	87%	2%	0%	0%	0%	0%	0%	0%	10%
Collab.	0%	1%	0%	84%	5%	2%	0%	0%	0%	0%	8%
Comm.	0%	1%	1%	13%	71%	2%	0%	0%	0%	0%	12%
Infra.	0%	1%	0%	3%	3%	88%	0%	1%	0%	0%	3%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	99%	0%
SNG	0%	0%	0%	2%	1%	0%	0%	0%	1%	0%	95%

Table A.15: All Instances (Automatic Partitioning) BioHEL-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	93%	0%	0%	0%	0%	3%	2%	1%	0%	0%	0%
A.Bio.	6%	6%	10%	25%	10%	6%	7%	2%	0%	8%	21%
Code	0%	5%	18%	16%	4%	7%	1%	6%	2%	4%	38%
Collab.	0%	7%	15%	10%	10%	13%	0%	9%	3%	10%	23%
Comm.	0%	4%	10%	17%	5%	13%	0%	3%	4%	10%	34%
Infra.	7%	3%	5%	11%	8%	26%	1%	18%	1%	5%	15%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	4%	0%	1%	1%	0%	8%	1%	75%	4%	3%	1%
Tweet	0%	0%	2%	1%	0%	1%	0%	4%	82%	0%	10%
RDG	1%	1%	3%	2%	0%	2%	0%	1%	0%	88%	2%
SNG	2%	4%	8%	7%	6%	9%	1%	4%	8%	4%	48%

Table A.16: Non Outliers (Automatic Partitioning) KNN-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	88%	0%	0%	0%	0%	11%	0%	1%	0%	0%	0%
A.Bio.	0%	21%	11%	20%	10%	14%	0%	22%	0%	0%	2%
Code	0%	2%	42%	4%	0%	10%	0%	10%	1%	10%	21%
Collab.	0%	18%	11%	9%	21%	2%	0%	0%	0%	10%	29%
Comm.	0%	2%	0%	11%	15%	7%	0%	0%	16%	1%	48%
Infra.	6%	15%	1%	1%	2%	38%	0%	17%	1%	0%	20%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	4%	3%	0%	3%	0%	12%	0%	76%	1%	0%	0%
Tweet	0%	0%	5%	0%	0%	0%	0%	5%	73%	0%	17%
RDG	0%	3%	4%	0%	0%	0%	0%	0%	0%	93%	0%
SNG	0%	3%	7%	10%	8%	3%	0%	0%	14%	3%	51%

Table A.17: Non Outliers (Automatic Partitioning) KNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	89%	0%	0%	0%	0%	11%	0%	0%	0%	0%	0%
A.Bio.	0%	22%	11%	22%	11%	11%	0%	22%	0%	0%	0%
Code	0%	0%	44%	0%	0%	11%	0%	11%	0%	11%	22%
Collab.	0%	30%	10%	10%	20%	0%	0%	0%	0%	10%	20%
Comm.	0%	0%	0%	8%	17%	8%	0%	0%	17%	0%	50%
Infra.	6%	17%	0%	0%	0%	39%	0%	17%	0%	0%	22%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	4%	4%	0%	4%	0%	11%	0%	74%	4%	0%	0%
Tweet	0%	0%	5%	0%	0%	0%	0%	5%	74%	0%	16%
RDG	0%	4%	4%	0%	0%	0%	0%	0%	0%	93%	0%
SNG	0%	7%	7%	7%	7%	3%	0%	0%	13%	3%	53%

Table A.18: Non Outliers (Automatic Partitioning) IKNN-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	32%	0%	0%	0%	0%	17%	0%	49%	2%	0%	0%
A.Bio.	5%	1%	9%	20%	10%	31%	0%	17%	1%	0%	6%
Code	0%	0%	30%	14%	0%	11%	0%	9%	4%	10%	23%
Collab.	0%	19%	10%	10%	13%	6%	0%	0%	0%	7%	36%
Comm.	0%	2%	2%	24%	0%	14%	0%	0%	9%	0%	49%
Infra.	3%	11%	2%	2%	2%	42%	0%	14%	2%	0%	22%
Meta.	0%	0%	0%	0%	0%	0%	10%	80%	0%	0%	10%
Social	2%	7%	2%	5%	1%	20%	0%	46%	13%	0%	5%
Tweet	0%	1%	4%	5%	9%	2%	0%	1%	39%	0%	39%
RDG	0%	4%	4%	12%	6%	0%	0%	0%	0%	48%	26%
SNG	0%	1%	8%	11%	28%	6%	0%	1%	19%	3%	23%

Table A.19: Non Outliers (Automatic Partitioning) IKNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	39%	0%	0%	0%	0%	11%	0%	44%	6%	0%	0%
A.Bio.	0%	22%	11%	22%	11%	11%	0%	22%	0%	0%	0%
Code	0%	0%	44%	0%	0%	11%	0%	11%	0%	11%	22%
Collab.	0%	30%	10%	10%	10%	10%	0%	0%	0%	0%	30%
Comm.	0%	0%	8%	8%	17%	17%	0%	0%	8%	0%	42%
Infra.	6%	6%	0%	0%	6%	44%	0%	11%	0%	0%	28%
Meta.	0%	0%	0%	0%	0%	0%	9%	79%	0%	0%	12%
Social	4%	4%	4%	7%	4%	11%	0%	59%	4%	0%	4%
Tweet	0%	0%	0%	5%	11%	0%	0%	5%	74%	0%	5%
RDG	0%	4%	4%	4%	4%	0%	0%	0%	0%	59%	26%
SNG	0%	3%	7%	3%	23%	7%	0%	0%	13%	3%	40%

Table A.20: Non Outliers (Automatic Partitioning) SVM-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	11%	12%	4%	5%	5%	11%	0%	6%	0%	1%	44%
Code	0%	0%	26%	4%	5%	14%	0%	0%	0%	10%	42%
Collab.	0%	0%	9%	17%	8%	10%	0%	8%	0%	10%	39%
Comm.	0%	0%	0%	6%	27%	4%	0%	0%	8%	1%	55%
Infra.	9%	1%	2%	1%	1%	60%	0%	2%	0%	1%	23%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	13%	0%	86%	0%	0%	0%
Tweet	0%	0%	1%	0%	0%	0%	0%	0%	94%	0%	5%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	0%	2%	1%	3%	6%	0%	0%	17%	3%	69%

Table A.21: Non Outliers (Automatic Partitioning) SVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	11%	0%	11%	0%	0%	22%	0%	11%	0%	0%	44%
Code	0%	0%	0%	0%	11%	11%	0%	0%	0%	11%	67%
Collab.	0%	0%	10%	0%	10%	20%	0%	10%	0%	10%	40%
Comm.	0%	0%	0%	8%	8%	8%	0%	0%	8%	8%	58%
Infra.	11%	0%	6%	0%	0%	39%	0%	11%	0%	6%	28%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	15%	0%	74%	4%	4%	4%
Tweet	0%	0%	5%	0%	0%	0%	0%	5%	74%	0%	16%
RDG	0%	0%	0%	0%	4%	0%	0%	0%	0%	96%	0%
SNG	0%	0%	3%	7%	7%	7%	0%	0%	23%	3%	50%

Table A.22: Non Outliers (Automatic Partitioning) NUSVM-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	6%	63%	1%	4%	6%	12%	0%	0%	0%	0%	7%
Code	0%	9%	73%	4%	0%	1%	0%	0%	0%	1%	12%
Collab.	0%	17%	23%	41%	9%	2%	0%	1%	0%	3%	3%
Comm.	0%	7%	13%	17%	47%	4%	0%	1%	0%	0%	11%
Infra.	2%	4%	4%	4%	1%	73%	0%	7%	0%	1%	4%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	1%	0%	3%	0%	95%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	93%	0%	7%
RDG	0%	0%	1%	0%	0%	0%	0%	0%	0%	99%	0%
SNG	0%	8%	7%	4%	12%	3%	0%	0%	14%	3%	48%

Table A.23: Non Outliers (Automatic Partitioning) NUSVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	11%	22%	11%	0%	22%	0%	0%	0%	11%	0%	22%
Code	0%	11%	33%	11%	0%	0%	0%	11%	0%	11%	22%
Collab.	0%	20%	40%	0%	0%	0%	0%	10%	0%	10%	20%
Comm.	0%	8%	17%	8%	33%	8%	0%	0%	8%	8%	8%
Infra.	6%	11%	6%	11%	0%	39%	0%	17%	0%	0%	11%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	15%	0%	0%	7%	4%	70%	0%	0%	4%
Tweet	0%	0%	0%	0%	0%	0%	0%	5%	68%	0%	26%
RDG	0%	4%	4%	0%	0%	0%	0%	0%	0%	93%	0%
SNG	0%	10%	20%	3%	20%	7%	0%	0%	27%	3%	10%

Table A.24: Non Outliers (Automatic Partitioning) BioHEL-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	74%	1%	9%	7%	0%	0%	0%	0%	0%	9%
Code	0%	0%	80%	3%	1%	0%	0%	0%	0%	0%	16%
Collab.	0%	0%	0%	86%	5%	0%	0%	0%	0%	0%	8%
Comm.	0%	0%	0%	6%	79%	0%	0%	0%	0%	0%	15%
Infra.	0%	0%	0%	1%	2%	92%	0%	2%	0%	0%	3%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	0%	0%	0%	1%	0%	0%	0%	1%	0%	97%

Table A.25: Non Outliers (Automatic Partitioning) BioHEL-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	95%	0%	0%	0%	0%	2%	2%	1%	0%	0%	0%
A.Bio.	8%	4%	14%	16%	9%	6%	7%	2%	0%	4%	31%
Code	0%	4%	17%	16%	6%	8%	0%	11%	1%	2%	35%
Collab.	0%	7%	15%	5%	15%	9%	0%	16%	4%	9%	19%
Comm.	0%	3%	9%	15%	7%	6%	0%	5%	6%	7%	42%
Infra.	8%	4%	4%	7%	4%	30%	0%	22%	0%	3%	18%
Meta.	0%	0%	0%	0%	0%	0%	99%	0%	0%	0%	0%
Social	4%	1%	2%	2%	1%	11%	2%	69%	6%	1%	2%
Tweet	0%	0%	3%	1%	0%	1%	0%	5%	80%	0%	10%
RDG	0%	1%	1%	1%	1%	1%	0%	0%	0%	92%	2%
SNG	0%	2%	8%	4%	9%	7%	2%	4%	8%	4%	52%

Table A.26: Non Outliers (70/30 Vertex based) KNN-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	92%	0%	0%	0%	0%	8%	0%	0%	0%	0%	0%
A.Bio.	0%	33%	0%	17%	0%	17%	0%	17%	17%	0%	0%
Code	0%	17%	50%	0%	0%	0%	0%	17%	0%	0%	17%
Collab.	0%	43%	0%	0%	29%	0%	0%	0%	0%	14%	14%
Comm.	0%	0%	0%	13%	0%	13%	0%	0%	25%	0%	50%
Infra.	0%	8%	0%	0%	17%	42%	0%	25%	8%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	6%	0%	0%	0%	0%	17%	0%	72%	0%	0%	6%
Tweet	0%	0%	0%	0%	0%	0%	0%	8%	69%	0%	23%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	5%	14%	10%	5%	0%	0%	0%	29%	5%	33%

Table A.27: Non Outliers (70/30 Vertex based) KNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	83%	0%	0%	0%	0%	17%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	33%	33%	0%	0%	33%	0%	0%	0%
Code	0%	0%	0%	33%	0%	0%	0%	0%	33%	0%	33%
Collab.	0%	0%	33%	0%	67%	0%	0%	0%	0%	0%	0%
Comm.	0%	0%	0%	0%	25%	0%	0%	0%	0%	0%	75%
Infra.	17%	33%	0%	0%	0%	33%	0%	17%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	11%	0%	0%	0%	0%	11%	78%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	83%	0%	17%
RDG	0%	11%	11%	0%	0%	0%	0%	0%	0%	78%	0%
SNG	0%	0%	0%	33%	11%	11%	0%	0%	0%	0%	44%

Table A.28: Non Outliers (70/30 Vertex based) IKNN-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	8%	0%	0%	0%	0%	17%	0%	75%	0%	0%	0%
A.Bio.	17%	0%	0%	17%	0%	17%	0%	0%	0%	0%	50%
Code	0%	0%	50%	0%	0%	0%	0%	17%	0%	0%	33%
Collab.	0%	29%	0%	0%	43%	0%	0%	0%	0%	0%	29%
Comm.	0%	0%	0%	13%	0%	25%	0%	0%	13%	0%	50%
Infra.	8%	8%	0%	0%	17%	33%	0%	25%	0%	0%	8%
Meta.	0%	0%	0%	0%	93%	0%	0%	0%	0%	0%	7%
Social	6%	0%	6%	0%	0%	33%	0%	28%	11%	0%	17%
Tweet	0%	0%	0%	15%	15%	0%	0%	0%	46%	0%	23%
RDG	0%	0%	0%	28%	6%	0%	0%	0%	0%	56%	11%
SNG	0%	0%	14%	10%	14%	0%	0%	0%	33%	5%	24%

Table A.29: Non Outliers (70/30 Vertex based) IKNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	17%	0%	83%	0%	0%	0%
A.Bio.	0%	0%	0%	33%	33%	0%	0%	33%	0%	0%	0%
Code	0%	0%	33%	33%	0%	0%	0%	0%	0%	0%	33%
Collab.	0%	0%	33%	0%	67%	0%	0%	0%	0%	0%	0%
Comm.	0%	0%	0%	0%	25%	0%	0%	0%	0%	0%	75%
Infra.	17%	17%	0%	0%	0%	50%	0%	0%	0%	0%	17%
Meta.	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%
Social	0%	11%	0%	0%	11%	22%	0%	44%	11%	0%	0%
Tweet	0%	0%	0%	0%	17%	0%	0%	0%	50%	0%	33%
RDG	0%	11%	11%	33%	0%	0%	0%	11%	0%	33%	0%
SNG	0%	0%	0%	33%	11%	11%	0%	0%	11%	0%	33%

Table A.30: Non Outliers (70/30 Vertex based) SVM-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	17%	33%	0%	17%	0%	0%	0%	0%	0%	0%	33%
Code	0%	0%	17%	17%	17%	0%	0%	0%	0%	0%	50%
Collab.	0%	0%	0%	43%	0%	14%	0%	14%	0%	14%	14%
Comm.	0%	0%	0%	13%	50%	0%	0%	0%	0%	0%	38%
Infra.	0%	0%	0%	8%	0%	75%	0%	8%	0%	0%	8%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	6%	0%	6%	0%	89%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	85%	0%	15%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	0%	0%	0%	5%	5%	0%	0%	5%	5%	81%

Table A.31: Non Outliers (70/30 Vertex based) SVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	33%	0%	0%	0%	33%	0%	0%	33%
Code	0%	0%	0%	0%	0%	33%	0%	0%	0%	0%	67%
Collab.	0%	0%	0%	0%	0%	33%	0%	0%	0%	0%	67%
Comm.	0%	0%	0%	0%	25%	0%	0%	0%	0%	0%	75%
Infra.	17%	0%	0%	17%	0%	33%	0%	17%	0%	0%	17%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	11%	11%	0%	11%	67%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	33%	0%	67%
RDG	0%	0%	0%	0%	0%	11%	0%	0%	0%	89%	0%
SNG	0%	0%	0%	11%	11%	22%	0%	0%	0%	0%	56%

Table A.32: Non Outliers (70/30 Vertex based) NUSVM-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Code	0%	0%	83%	0%	17%	0%	0%	0%	0%	0%	0%
Collab.	0%	29%	14%	43%	14%	0%	0%	0%	0%	0%	0%
Comm.	0%	0%	13%	13%	75%	0%	0%	0%	0%	0%	0%
Infra.	0%	8%	8%	0%	0%	75%	0%	8%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	6%	0%	0%	0%	0%	6%	0%	89%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	0%	10%	5%	29%	5%	0%	0%	24%	5%	24%

Table A.33: Non Outliers (70/30 Vertex based) NUSVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	33%	0%	0%	33%	0%	0%	0%	33%
Code	0%	0%	0%	33%	0%	33%	0%	0%	0%	0%	33%
Collab.	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Comm.	0%	25%	0%	0%	75%	0%	0%	0%	0%	0%	0%
Infra.	17%	17%	0%	17%	0%	33%	0%	17%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	22%	0%	0%	22%	56%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	67%	0%	33%
RDG	0%	11%	11%	0%	0%	0%	0%	0%	0%	78%	0%
SNG	0%	11%	22%	0%	11%	22%	0%	0%	11%	0%	22%

Table A.34: Non Outliers (70/30 Vertex based) BioHEL-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	0%	0%	86%	0%	0%	0%	0%	0%	0%	14%
Comm.	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%
Infra.	0%	0%	0%	0%	0%	83%	0%	17%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%

Table A.35: Non Outliers (70/30 Vertex based) BioHEL-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	33%	0%	0%	0%	0%	0%	0%	67%
Code	0%	0%	67%	0%	0%	33%	0%	0%	0%	0%	0%
Collab.	0%	33%	0%	67%	0%	0%	0%	0%	0%	0%	0%
Comm.	0%	0%	0%	0%	0%	0%	0%	0%	0%	25%	75%
Infra.	17%	17%	0%	50%	0%	0%	0%	17%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	0%	33%	56%	0%	11%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%
RDG	0%	11%	0%	0%	0%	0%	0%	0%	0%	89%	0%
SNG	0%	0%	0%	11%	11%	11%	11%	0%	11%	0%	44%

Table A.36: Non Outliers (70/30 Edge based) KNN-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	92%	0%	0%	0%	0%	8%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	17%	0%	17%	33%	0%	17%	0%	0%	17%
Code	0%	17%	33%	0%	0%	0%	0%	17%	17%	0%	17%
Collab.	0%	14%	14%	14%	14%	14%	0%	0%	0%	0%	29%
Comm.	0%	13%	0%	0%	0%	25%	0%	0%	25%	0%	38%
Infra.	0%	17%	0%	0%	0%	25%	0%	25%	0%	0%	33%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	6%	0%	0%	0%	6%	6%	78%	0%	0%	6%
Tweet	0%	0%	8%	0%	0%	0%	0%	8%	62%	0%	23%
RDG	0%	6%	6%	0%	0%	0%	0%	0%	0%	89%	0%
SNG	0%	5%	5%	14%	5%	5%	0%	0%	10%	5%	52%

Table A.37: Non Outliers (70/30 Edge based) KNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	83%	0%	0%	0%	0%	17%	0%	0%	0%	0%	0%
A.Bio.	0%	33%	0%	33%	0%	0%	0%	33%	0%	0%	0%
Code	0%	0%	67%	0%	0%	0%	0%	0%	0%	33%	0%
Collab.	0%	33%	0%	0%	33%	0%	0%	0%	0%	33%	0%
Comm.	0%	25%	0%	0%	25%	0%	0%	0%	0%	25%	25%
Infra.	17%	17%	0%	0%	0%	50%	0%	17%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	22%	0%	0%	0%	0%	11%	0%	67%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	0%	11%	11%	22%	0%	0%	0%	11%	0%	44%

Table A.38: Non Outliers (70/30 Edge based) IKNN-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	17%	0%	0%	0%	0%	8%	0%	75%	0%	0%	0%
A.Bio.	17%	0%	17%	0%	17%	33%	0%	0%	0%	0%	17%
Code	0%	17%	33%	0%	0%	0%	0%	17%	17%	0%	17%
Collab.	0%	14%	0%	14%	14%	14%	0%	0%	0%	14%	29%
Comm.	0%	13%	0%	0%	0%	38%	0%	0%	0%	0%	50%
Infra.	0%	17%	17%	0%	0%	25%	0%	8%	8%	0%	25%
Meta.	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%
Social	6%	11%	11%	0%	0%	11%	0%	44%	11%	0%	6%
Tweet	0%	0%	8%	8%	8%	0%	0%	0%	38%	0%	38%
RDG	0%	17%	6%	0%	0%	0%	0%	0%	0%	39%	39%
SNG	0%	5%	5%	14%	24%	5%	0%	0%	10%	5%	33%

Table A.39: Non Outliers (70/30 Edge based) IKNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	17%	0%	0%	0%	0%	0%	0%	83%	0%	0%	0%
A.Bio.	0%	33%	0%	33%	0%	0%	0%	33%	0%	0%	0%
Code	0%	0%	67%	0%	0%	0%	0%	0%	0%	33%	0%
Collab.	0%	33%	0%	0%	33%	33%	0%	0%	0%	0%	0%
Comm.	0%	25%	25%	0%	25%	0%	0%	0%	0%	0%	25%
Infra.	17%	17%	17%	0%	0%	50%	0%	0%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%
Social	22%	0%	11%	0%	0%	11%	0%	44%	11%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	50%	0%	50%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	67%	33%
SNG	0%	0%	11%	11%	33%	0%	0%	0%	11%	0%	33%

Table A.40: Non Outliers (70/30 Edge based) SVM-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	17%	33%	0%	0%	0%	17%	0%	0%	0%	0%	33%
Code	0%	0%	17%	0%	0%	33%	0%	0%	0%	0%	50%
Collab.	0%	0%	0%	57%	0%	0%	0%	14%	0%	0%	29%
Comm.	0%	0%	0%	13%	13%	0%	0%	0%	0%	0%	75%
Infra.	8%	0%	0%	8%	0%	67%	0%	0%	0%	0%	17%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	11%	0%	89%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	77%	0%	23%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	0%	0%	0%	0%	5%	0%	0%	0%	5%	90%

Table A.41: Non Outliers (70/30 Edge based) SVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	33%	0%	0%	0%	33%	0%	0%	33%
Code	0%	0%	33%	0%	0%	0%	0%	0%	0%	33%	33%
Collab.	0%	0%	0%	0%	0%	33%	0%	0%	0%	33%	33%
Comm.	0%	0%	0%	25%	0%	0%	0%	0%	0%	25%	50%
Infra.	17%	0%	17%	0%	0%	50%	0%	0%	0%	0%	17%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%
Tweet	0%	0%	0%	17%	0%	0%	0%	0%	33%	0%	50%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	0%	0%	11%	11%	0%	0%	0%	22%	0%	56%

Table A.42: Non Outliers (70/30 Edge based) NUSVM-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Code	0%	17%	83%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	0%	14%	86%	0%	0%	0%	0%	0%	0%	0%
Comm.	0%	0%	13%	13%	63%	0%	0%	0%	0%	0%	13%
Infra.	0%	8%	8%	0%	8%	67%	0%	8%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	6%	0%	94%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	77%	0%	23%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	5%	0%	0%	14%	0%	0%	0%	0%	5%	76%

Table A.43: Non Outliers (70/30 Edge based) NUSVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	33%	33%	33%	0%	0%	0%	0%	0%	0%	0%
Code	0%	33%	0%	33%	0%	0%	0%	0%	0%	33%	0%
Collab.	0%	0%	33%	0%	33%	0%	0%	0%	0%	33%	0%
Comm.	0%	0%	75%	0%	25%	0%	0%	0%	0%	0%	0%
Infra.	0%	0%	33%	0%	0%	67%	0%	0%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	22%	0%	0%	0%	0%	11%	11%	56%	0%	0%	0%
Tweet	0%	0%	17%	0%	0%	0%	0%	0%	83%	0%	0%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	11%	11%	0%	33%	0%	0%	0%	11%	0%	33%

Table A.44: Non Outliers (70/30 Edge based) BioHEL-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	86%	3%	1%	10%	0%	0%	0%	0%	0%	0%
Code	0%	0%	98%	0%	0%	0%	0%	0%	0%	0%	1%
Collab.	0%	0%	0%	97%	0%	1%	0%	1%	0%	0%	0%
Comm.	0%	3%	1%	5%	87%	0%	0%	0%	0%	0%	4%
Infra.	0%	1%	1%	1%	5%	87%	0%	5%	0%	0%	2%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	99%

Table A.45: Non Outliers (70/30 Edge based) BioHEL-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	18%	14%	25%	10%	5%	11%	5%	0%	3%	9%
Code	0%	9%	42%	15%	2%	6%	3%	12%	0%	6%	5%
Collab.	0%	7%	8%	1%	9%	11%	0%	11%	1%	29%	23%
Comm.	0%	3%	20%	3%	18%	8%	0%	10%	0%	15%	24%
Infra.	17%	4%	8%	5%	5%	28%	0%	16%	0%	3%	14%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	20%	0%	0%	0%	0%	7%	6%	63%	3%	0%	0%
Tweet	0%	0%	1%	0%	0%	1%	0%	3%	85%	0%	9%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	2%	16%	7%	4%	4%	3%	6%	15%	0%	43%

Table A.46: Non Outliers (90/10 Vertex based) KNN-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	88%	0%	0%	0%	0%	13%	0%	0%	0%	0%	0%
A.Bio.	0%	25%	13%	13%	13%	13%	0%	25%	0%	0%	0%
Code	0%	13%	25%	0%	0%	0%	0%	13%	13%	13%	25%
Collab.	0%	22%	11%	11%	22%	0%	0%	0%	0%	11%	22%
Comm.	0%	0%	0%	0%	20%	10%	0%	0%	0%	0%	70%
Infra.	0%	13%	0%	0%	0%	44%	0%	19%	0%	0%	25%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	4%	0%	0%	4%	0%	13%	0%	75%	4%	0%	0%
Tweet	0%	0%	6%	0%	0%	0%	0%	6%	71%	0%	18%
RDG	0%	0%	4%	0%	0%	0%	0%	0%	0%	96%	0%
SNG	0%	4%	7%	11%	7%	4%	0%	0%	11%	4%	52%

Table A.47: Non Outliers (90/10 Vertex based) KNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%
Comm.	0%	0%	0%	50%	0%	0%	0%	0%	50%	0%	0%
Infra.	50%	50%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	33%	0%	0%	0%	0%	0%	67%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%
RDG	0%	33%	0%	0%	0%	0%	0%	0%	0%	67%	0%
SNG	0%	0%	0%	33%	0%	0%	0%	0%	33%	0%	33%

Table A.48: Non Outliers (90/10 Vertex based) IKNN-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	25%	0%	75%	0%	0%	0%
A.Bio.	0%	0%	13%	13%	13%	13%	0%	25%	0%	0%	25%
Code	0%	0%	25%	0%	0%	0%	0%	13%	25%	13%	25%
Collab.	0%	11%	11%	22%	0%	0%	0%	0%	0%	11%	44%
Comm.	0%	0%	10%	20%	0%	10%	0%	0%	10%	0%	50%
Infra.	0%	13%	0%	0%	0%	50%	0%	13%	0%	0%	25%
Meta.	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%
Social	0%	4%	0%	8%	0%	25%	0%	50%	8%	0%	4%
Tweet	0%	0%	6%	0%	0%	0%	0%	6%	53%	0%	35%
RDG	0%	0%	4%	4%	0%	0%	0%	0%	0%	54%	38%
SNG	0%	0%	7%	11%	30%	4%	0%	0%	22%	4%	22%

Table A.49: Non Outliers (90/10 Vertex based) IKNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%
A.Bio.	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%
Comm.	0%	0%	0%	50%	0%	50%	0%	0%	0%	0%	0%
Infra.	0%	0%	0%	0%	0%	50%	0%	0%	0%	0%	50%
Meta.	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%
Social	0%	33%	0%	0%	0%	0%	0%	67%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%
RDG	0%	33%	0%	0%	0%	0%	0%	0%	0%	67%	0%
SNG	0%	0%	0%	33%	33%	0%	0%	0%	0%	0%	33%

Table A.50: Non Outliers (90/10 Vertex based) SVM-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	13%	13%	13%	0%	0%	13%	0%	0%	0%	0%	50%
Code	0%	0%	13%	0%	0%	38%	0%	0%	0%	0%	50%
Collab.	0%	0%	0%	56%	0%	0%	0%	11%	0%	11%	22%
Comm.	0%	0%	0%	0%	30%	10%	0%	0%	0%	0%	60%
Infra.	0%	0%	6%	0%	0%	69%	0%	6%	0%	0%	19%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	13%	0%	88%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	0%	0%	4%	7%	4%	0%	0%	19%	4%	63%

Table A.51: Non Outliers (90/10 Vertex based) SVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%
Code	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Collab.	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Comm.	0%	0%	0%	0%	0%	0%	0%	0%	50%	50%	0%
Infra.	50%	0%	0%	0%	0%	0%	0%	0%	0%	0%	50%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	33%	0%	67%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	0%	0%	33%	33%	0%	0%	0%	0%	0%	33%

Table A.52: Non Outliers (90/10 Vertex based) NUSVM-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	13%	50%	0%	13%	0%	13%	0%	13%	0%	0%	0%
Code	0%	13%	63%	0%	0%	13%	0%	0%	0%	0%	13%
Collab.	0%	0%	11%	56%	22%	11%	0%	0%	0%	0%	0%
Comm.	0%	0%	10%	0%	60%	10%	0%	0%	0%	0%	20%
Infra.	0%	6%	0%	6%	0%	63%	0%	25%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	4%	0%	96%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	94%	0%	6%
RDG	0%	0%	4%	0%	0%	0%	0%	0%	0%	96%	0%
SNG	0%	11%	0%	4%	4%	7%	0%	0%	26%	4%	44%

Table A.53: Non Outliers (90/10 Vertex based) NUSVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%
Comm.	0%	0%	0%	0%	0%	0%	0%	0%	0%	50%	50%
Infra.	50%	0%	0%	0%	0%	50%	0%	0%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	33%	0%	0%	67%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	33%	33%	0%	0%	0%	0%	0%	0%	0%	33%

Table A.54: Non Outliers (90/10 Vertex based) BioHEL-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	78%	1%	14%	5%	0%	0%	0%	0%	0%	2%
Code	0%	1%	75%	13%	1%	1%	0%	0%	0%	0%	9%
Collab.	0%	0%	0%	93%	2%	0%	0%	0%	0%	0%	4%
Comm.	0%	0%	0%	12%	80%	1%	0%	0%	0%	0%	6%
Infra.	0%	0%	0%	1%	3%	94%	0%	2%	0%	0%	1%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	0%	0%	1%	1%	0%	0%	0%	1%	0%	97%

Table A.55: Non Outliers (90/10 Vertex based) BioHEL-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	99%	0%	0%	0%	0%	0%	1%	0%	0%	0%	0%
A.Bio.	0%	2%	6%	83%	0%	2%	0%	2%	0%	1%	5%
Code	0%	12%	58%	4%	2%	5%	0%	0%	0%	0%	20%
Collab.	0%	2%	34%	12%	6%	26%	2%	0%	0%	0%	18%
Comm.	0%	1%	9%	34%	7%	5%	0%	1%	12%	3%	29%
Infra.	47%	5%	6%	22%	4%	7%	0%	2%	0%	0%	7%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	1%	1%	4%	15%	5%	4%	4%	63%	0%	2%	1%
Tweet	0%	0%	0%	0%	0%	0%	0%	1%	99%	0%	0%
RDG	0%	4%	7%	10%	3%	4%	0%	1%	0%	69%	2%
SNG	0%	1%	6%	17%	5%	2%	2%	19%	27%	0%	22%

Table A.56: Non Outliers (90/10 Edge based) KNN-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	88%	0%	0%	0%	0%	13%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	13%	25%	13%	13%	0%	25%	0%	0%	13%
Code	0%	13%	25%	0%	0%	0%	0%	13%	13%	13%	25%
Collab.	0%	22%	11%	11%	22%	0%	0%	0%	0%	0%	33%
Comm.	0%	10%	0%	30%	0%	10%	0%	0%	10%	0%	40%
Infra.	6%	13%	0%	0%	0%	44%	0%	13%	6%	0%	19%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	4%	4%	0%	4%	0%	17%	0%	71%	0%	0%	0%
Tweet	0%	0%	6%	0%	0%	0%	0%	6%	71%	0%	18%
RDG	0%	4%	4%	0%	0%	0%	0%	0%	0%	92%	0%
SNG	0%	4%	11%	11%	4%	4%	0%	0%	15%	4%	48%

Table A.57: Non Outliers (90/10 Edge based) KNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
Comm.	0%	0%	0%	0%	50%	0%	0%	0%	0%	0%	50%
Infra.	0%	50%	0%	0%	0%	50%	0%	0%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	67%	0%	33%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	0%	0%	33%	33%	0%	0%	0%	0%	0%	33%

Table A.58: Non Outliers (90/10 Edge based) IKNN-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	25%	0%	75%	0%	0%	0%
A.Bio.	0%	0%	13%	25%	13%	13%	0%	13%	0%	0%	25%
Code	0%	13%	25%	0%	0%	0%	0%	13%	13%	13%	25%
Collab.	0%	22%	11%	11%	22%	0%	0%	0%	0%	0%	33%
Comm.	0%	10%	0%	30%	0%	20%	0%	0%	10%	0%	30%
Infra.	0%	13%	0%	0%	0%	44%	0%	19%	0%	0%	25%
Meta.	0%	0%	0%	0%	3%	0%	0%	0%	0%	0%	97%
Social	0%	8%	0%	4%	17%	21%	0%	29%	21%	0%	0%
Tweet	0%	0%	6%	0%	12%	0%	0%	0%	41%	0%	41%
RDG	0%	13%	4%	0%	8%	0%	0%	0%	0%	38%	38%
SNG	0%	4%	11%	11%	26%	4%	0%	0%	19%	4%	22%

Table A.59: Non Outliers (90/10 Edge based) IKNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%
A.Bio.	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%
Comm.	0%	0%	50%	0%	50%	0%	0%	0%	0%	0%	0%
Infra.	0%	50%	0%	0%	0%	50%	0%	0%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Social	0%	0%	0%	0%	0%	67%	0%	33%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	0%	0%	33%	33%	0%	0%	0%	0%	0%	33%

Table A.60: Non Outliers (90/10 Edge based) SVM-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	13%	13%	0%	13%	0%	13%	0%	13%	0%	0%	38%
Code	0%	0%	13%	13%	0%	13%	0%	0%	0%	13%	50%
Collab.	0%	0%	11%	33%	11%	0%	0%	11%	0%	0%	33%
Comm.	0%	0%	0%	10%	50%	10%	0%	0%	10%	0%	20%
Infra.	6%	0%	0%	0%	0%	69%	0%	0%	0%	0%	25%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	17%	0%	83%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	94%	0%	6%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	0%	0%	4%	4%	4%	0%	0%	19%	4%	67%

Table A.61: Non Outliers (90/10 Edge based) SVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Code	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Collab.	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
Comm.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Infra.	0%	0%	0%	0%	0%	50%	0%	0%	0%	0%	50%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	0%	0%	0%	33%	0%	0%	0%	0%	0%	67%

Table A.62: Non Outliers (90/10 Edge based) NUSVM-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	13%	38%	0%	0%	25%	0%	0%	0%	0%	0%	25%
Code	0%	13%	75%	0%	0%	13%	0%	0%	0%	0%	0%
Collab.	0%	11%	22%	44%	11%	11%	0%	0%	0%	0%	0%
Comm.	0%	0%	0%	30%	50%	0%	0%	0%	0%	0%	20%
Infra.	0%	6%	13%	0%	6%	75%	0%	0%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	4%	0%	0%	4%	0%	92%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%
RDG	0%	0%	4%	0%	0%	0%	0%	0%	0%	96%	0%
SNG	0%	4%	7%	0%	0%	4%	0%	0%	19%	4%	63%

Table A.63: Non Outliers (90/10 Edge based) NUSVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
Comm.	0%	0%	50%	50%	0%	0%	0%	0%	0%	0%	0%
Infra.	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	33%	0%	67%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	0%	0%	33%	0%	0%	0%	0%	0%	0%	67%

Table A.64: Non Outliers (90/10 Edge based) BioHEL-C Training Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	72%	2%	7%	11%	1%	0%	0%	0%	0%	8%
Code	0%	0%	76%	5%	3%	1%	0%	0%	0%	0%	15%
Collab.	0%	0%	0%	89%	4%	0%	0%	1%	0%	0%	6%
Comm.	0%	0%	0%	6%	85%	0%	0%	0%	0%	0%	8%
Infra.	0%	1%	0%	0%	4%	94%	0%	0%	0%	0%	1%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%
SNG	0%	0%	0%	0%	0%	0%	0%	0%	1%	0%	98%

Table A.65: Non Outliers (90/10 Edge based) BioHEL-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	99%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	3%	8%	34%	22%	1%	0%	0%	0%	0%	31%
Code	0%	8%	34%	14%	2%	4%	0%	1%	0%	1%	36%
Collab.	0%	13%	2%	0%	0%	3%	0%	3%	0%	77%	2%
Comm.	0%	2%	7%	33%	7%	1%	0%	4%	0%	0%	48%
Infra.	3%	1%	4%	16%	3%	46%	0%	1%	0%	3%	24%
Meta.	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%
Social	0%	0%	1%	0%	0%	62%	0%	36%	0%	1%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	3%	68%	0%	29%
RDG	0%	0%	0%	0%	0%	0%	0%	0%	0%	99%	1%
SNG	0%	1%	6%	5%	16%	0%	0%	1%	1%	1%	68%

Table A.66: Non Outliers Vs Outliers (Automatic Partitioning) KNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	0%	0%	0%	0%	67%	0%	0%	0%	0%	33%
Comm.	0%	0%	0%	0%	0%	50%	0%	0%	0%	0%	50%
Infra.	0%	25%	0%	0%	50%	25%	0%	0%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	33%	33%	0%	0%	0%	0%	0%	0%	0%	33%	0%
SNG	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Table A.67: Non Outliers Vs Outliers (Automatic Partitioning) IKNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	0%	0%	0%	0%	67%	0%	0%	0%	0%	33%
Comm.	0%	0%	0%	0%	50%	50%	0%	0%	0%	0%	0%
Infra.	0%	25%	0%	0%	50%	25%	0%	0%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	33%	0%	67%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	0%	33%	0%	33%	0%	0%	0%	0%	0%	0%	33%
SNG	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%

Table A.68: Non Outliers Vs Outliers (Automatic Partitioning) SVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%
Code	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Collab.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Comm.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Infra.	0%	0%	0%	0%	0%	0%	25%	0%	0%	0%	75%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	33%	0%	67%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	33%	0%	0%	0%	0%	33%	0%	0%	0%	33%	0%
SNG	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Table A.69: Non Outliers Vs Outliers (Automatic Partitioning) NUSVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	33%	0%	0%	0%	67%	0%	0%	0%	0%	0%
Comm.	0%	0%	0%	0%	0%	50%	0%	0%	0%	0%	50%
Infra.	0%	0%	0%	50%	0%	0%	25%	0%	0%	0%	25%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	0%	0%	0%	0%	0%	33%	0%	33%	0%	33%	0%
SNG	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Table A.70: Non Outliers Vs Outliers (Automatic Partitioning) BioHEL-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	12%	3%	6%	25%	10%	0%	0%	0%	1%	42%
Code	0%	0%	9%	2%	9%	0%	0%	0%	0%	0%	79%
Collab.	0%	3%	14%	1%	4%	27%	6%	0%	0%	1%	43%
Comm.	0%	1%	2%	1%	22%	36%	0%	0%	0%	0%	37%
Infra.	0%	6%	3%	4%	39%	11%	3%	0%	0%	9%	25%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	10%	0%	0%	0%	0%	9%	0%	79%	1%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	6%	4%	6%	16%	1%	1%	0%	3%	0%	63%	0%
SNG	81%	0%	0%	1%	0%	17%	0%	0%	0%	0%	0%

Table A.71: Vertex Based 70 vs Outliers KNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	0%	0%	33%	0%	67%	0%	0%	0%	0%	0%
Comm.	0%	0%	0%	0%	0%	50%	0%	0%	0%	0%	50%
Infra.	0%	0%	0%	0%	50%	25%	25%	0%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	0%	0%	0%	0%	33%	0%	0%	67%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	33%	33%	0%	0%	0%	0%	0%	0%	0%	33%	0%
SNG	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Table A.72: Vertex Based 70 vs Outliers IKNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	0%	0%	33%	0%	67%	0%	0%	0%	0%	0%
Comm.	0%	0%	0%	0%	0%	50%	0%	0%	0%	0%	50%
Infra.	0%	0%	0%	0%	50%	25%	0%	0%	0%	0%	25%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	0%	0%	0%	0%	33%	67%	0%	0%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	0%	33%	0%	33%	0%	0%	0%	0%	0%	33%	0%
SNG	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Table A.73: Vertex Based 70 vs Outliers SVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%
Code	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Collab.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Comm.	0%	0%	0%	0%	0%	50%	0%	0%	0%	0%	50%
Infra.	0%	0%	0%	0%	50%	0%	25%	0%	0%	0%	25%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	0%	0%	0%	0%	33%	33%	0%	33%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	33%	0%	0%	33%	0%	0%	0%	0%	0%	33%	0%
SNG	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Table A.74: Vertex Based 70 vs Outliers NUSVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%
Code	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%
Collab.	0%	33%	0%	0%	0%	67%	0%	0%	0%	0%	0%
Comm.	0%	0%	0%	0%	0%	50%	0%	0%	0%	0%	50%
Infra.	0%	0%	0%	0%	50%	25%	25%	0%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	33%	0%	33%	0%	0%	0%	0%	0%	0%	33%	0%
SNG	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Table A.75: Vertex Based 70 vs Outliers BioHEL-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	4%	2%	11%	66%	4%	2%	0%	0%	0%	10%
Code	0%	0%	2%	0%	1%	12%	0%	0%	0%	0%	84%
Collab.	0%	3%	5%	1%	11%	44%	15%	0%	0%	0%	20%
Comm.	0%	0%	1%	2%	8%	44%	0%	0%	0%	0%	43%
Infra.	0%	1%	5%	2%	41%	21%	23%	0%	0%	0%	7%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	13%	0%	0%	0%	0%	7%	0%	78%	1%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	4%	7%	9%	12%	2%	1%	0%	1%	0%	64%	1%
SNG	84%	2%	0%	2%	1%	7%	0%	3%	0%	0%	0%

Table A.76: Edge Based 70 vs Outliers KNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	0%	0%	0%	0%	67%	0%	0%	0%	0%	33%
Comm.	0%	0%	0%	0%	50%	50%	0%	0%	0%	0%	0%
Infra.	0%	0%	0%	0%	25%	25%	25%	0%	0%	0%	25%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	33%	33%	0%	0%	0%	0%	0%	0%	0%	33%	0%
SNG	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Table A.77: Edge Based 70 vs Outliers IKNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	0%	0%	0%	0%	67%	0%	0%	0%	0%	33%
Comm.	0%	0%	0%	0%	50%	50%	0%	0%	0%	0%	0%
Infra.	0%	0%	0%	0%	25%	25%	0%	0%	0%	0%	50%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	0%	0%	33%	0%	33%	0%	0%	33%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	0%	33%	0%	0%	0%	0%	0%	33%	0%	0%	33%
SNG	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Table A.78: Edge Based 70 vs Outliers SVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Code	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Collab.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Comm.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Infra.	0%	0%	0%	0%	0%	0%	25%	0%	0%	0%	75%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	33%	0%	67%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	33%	0%	0%	0%	0%	33%	0%	0%	0%	33%	0%
SNG	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Table A.79: Edge Based 70 vs Outliers NUSVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	67%	0%	0%	0%	0%	0%	0%	0%	0%	33%
Comm.	0%	0%	0%	50%	0%	0%	0%	0%	0%	0%	50%
Infra.	0%	0%	0%	0%	50%	0%	25%	0%	0%	0%	25%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	0%	0%	0%	0%	33%	0%	0%	33%	0%	0%	33%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	0%	0%	0%	0%	0%	0%	0%	67%	0%	33%	0%
SNG	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Table A.80: Edge Based 70 vs Outliers BioHEL-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	4%	0%	0%	7%	6%	2%	2%	0%	10%	68%
Code	0%	1%	14%	2%	2%	3%	0%	0%	0%	1%	77%
Collab.	0%	2%	6%	5%	6%	7%	11%	0%	0%	1%	62%
Comm.	0%	2%	11%	2%	9%	14%	1%	0%	0%	6%	56%
Infra.	0%	1%	11%	2%	13%	9%	9%	0%	0%	11%	43%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	28%	0%	0%	0%	0%	2%	0%	67%	1%	0%	1%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	8%	10%	5%	0%	1%	3%	0%	6%	0%	66%	1%
SNG	82%	1%	1%	0%	1%	12%	0%	2%	0%	0%	0%

Table A.81: Vertex Based 90 vs Outliers KNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	0%	0%	0%	0%	67%	0%	0%	0%	0%	33%
Comm.	0%	0%	0%	0%	50%	50%	0%	0%	0%	0%	0%
Infra.	0%	25%	0%	0%	50%	25%	0%	0%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	33%	33%	0%	0%	0%	0%	0%	0%	0%	33%	0%
SNG	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Table A.82: Vertex Based 90 vs Outliers IKNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	0%	0%	0%	0%	67%	0%	0%	0%	0%	33%
Comm.	0%	0%	0%	0%	50%	50%	0%	0%	0%	0%	0%
Infra.	0%	25%	0%	0%	50%	25%	0%	0%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	33%	0%	67%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	0%	33%	0%	33%	0%	0%	0%	0%	0%	0%	33%
SNG	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Table A.83: Vertex Based 90 vs Outliers SVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%
Code	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Collab.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Comm.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Infra.	0%	0%	0%	0%	0%	0%	25%	0%	0%	0%	75%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	33%	0%	67%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	33%	0%	0%	0%	0%	33%	0%	0%	0%	33%	0%
SNG	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Table A.84: Vertex Based 90 vs Outliers NUSVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	33%	0%	0%	0%	67%	0%	0%	0%	0%	0%
Comm.	0%	0%	0%	0%	0%	50%	0%	0%	0%	0%	50%
Infra.	0%	25%	0%	0%	50%	0%	25%	0%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	0%	0%	33%	0%	0%	0%	0%	33%	0%	33%	0%
SNG	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Table A.85: Vertex Based 90 vs Outliers BioHEL-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	9%	2%	0%	18%	5%	0%	0%	0%	2%	63%
Code	0%	1%	6%	2%	8%	5%	0%	0%	0%	0%	78%
Collab.	0%	4%	16%	1%	6%	25%	6%	0%	0%	0%	42%
Comm.	0%	0%	3%	0%	19%	36%	0%	0%	0%	0%	41%
Infra.	0%	6%	5%	6%	39%	13%	4%	0%	0%	0%	26%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	12%	0%	0%	0%	0%	8%	0%	79%	1%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	4%	4%	4%	22%	0%	1%	0%	2%	0%	63%	0%
SNG	82%	0%	0%	1%	0%	16%	0%	0%	0%	0%	0%

Table A.86: Edge Based 90 vs Outliers KNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	0%	0%	0%	0%	67%	0%	0%	0%	0%	33%
Comm.	0%	0%	0%	0%	0%	50%	0%	0%	0%	0%	50%
Infra.	0%	25%	0%	0%	50%	25%	0%	0%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	33%	0%	67%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	33%	33%	0%	0%	0%	0%	0%	0%	0%	33%	0%
SNG	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Table A.87: Edge Based 90 vs Outliers IKNN-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	0%	0%	0%	0%	67%	0%	0%	0%	0%	33%
Comm.	0%	0%	0%	0%	0%	50%	0%	0%	0%	0%	50%
Infra.	0%	25%	0%	0%	50%	25%	0%	0%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	0%	0%	0%	33%	0%	33%	0%	33%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	0%	33%	0%	0%	0%	0%	0%	33%	0%	0%	33%
SNG	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Table A.88: Edge Based 90 vs Outliers SVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%
Code	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Collab.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Comm.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Infra.	0%	0%	0%	0%	25%	0%	25%	0%	0%	0%	50%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	67%	0%	33%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	33%	0%	0%	0%	0%	33%	0%	0%	0%	33%	0%
SNG	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Table A.89: Edge Based 90 vs Outliers NUSVM-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%
Code	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%
Collab.	0%	33%	0%	0%	0%	0%	0%	0%	0%	0%	67%
Comm.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%
Infra.	0%	0%	0%	50%	25%	0%	25%	0%	0%	0%	0%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	0%	0%	0%	0%	0%	33%	0%	67%	0%	0%	0%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	0%	0%	0%	0%	0%	0%	0%	67%	0%	33%	0%
SNG	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Table A.90: Edge Based 90 vs Outliers BioHEL-C Testing Set Confusion Matrix

	C.DNA	A.Bio.	Code	Collab.	Comm.	Infra.	Meta.	Social	Tweet	RDG	SNG
C.DNA	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
A.Bio.	0%	17%	6%	3%	33%	11%	0%	0%	0%	0%	29%
Code	0%	0%	12%	6%	11%	4%	0%	0%	0%	0%	67%
Collab.	0%	4%	10%	2%	4%	17%	4%	0%	0%	1%	57%
Comm.	0%	0%	3%	1%	21%	24%	0%	0%	0%	0%	51%
Infra.	0%	5%	3%	4%	36%	10%	3%	1%	0%	9%	29%
Meta.	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Social	10%	0%	0%	1%	0%	29%	0%	57%	1%	1%	1%
Tweet	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
RDG	6%	11%	7%	5%	1%	2%	0%	4%	0%	64%	0%
SNG	73%	2%	0%	0%	0%	20%	0%	4%	0%	0%	0%

Bibliography

[1] A. Abraham and A.E. Hassanien. *Computational Social Networks: Tools, Perspectives and Applications*. Springer London, 2012.

[2] S. Achard, C. Delon-Martin, P.E. Vértes, F. Renard, M. Schenck, F. Schneider, C. Heinrich, S. Kremer, and E.T. Bullmore. Hubs of brain functional networks are radically reorganized in comatose patients. *Proceedings of the National Academy of Sciences*, 109(50):20608–20613, 2012.

[3] S. Adali, X. Lu, and M. Magdon-Ismail. Attentive betweenness centrality (abc): Considering options and bandwidth when measuring criticality. In *2012 ASE/IEEE International Conference on Social Computing and 2012 ASE/IEEE International Conference on Privacy, Security, Risk and Trust*, pages 358–367, 2012. http://www.odysci.com/article/1010113018748685.

[4] L.A. Adamic and N. Glance. The political blogosphere and the 2004 us election: divided they blog. In *Proceedings of the 3rd international workshop on Link discovery*, pages 36–43. ACM, 2005.

[5] Y.-Y. Ahn, S. Han, H. Kwak, S. Moon, and H. Jeong. Analysis of topological characteristics of huge online social networking services. In *Proceedings of the 16th international conference on World Wide Web*, pages 835–844. ACM, 2007.

[6] J. Alcalá-Fdez, L. Sánchez, S. García, M.J. del Jesus, S. Ventura, J.M. Garrell, J. Otero, C. Romero, J. Bacardit, V.M. Rivas, J.C. Fernández, and F. Herrera. Keel: a software tool to assess evolutionary algorithms for data mining problems. *Soft Computing*, 13(3):307–318, 2009.

[7] D.R. Amancio, M.G.V. Nunes, O.N. Oliveira, and L. da Fontoura Costa. Extractive summarization using complex networks and syntactic dependency. *Physica A: Statistical Mechanics and its Applications*, 2011.

[8] J. Bacardit. *Pittsburgh Genetics-Based Machine Learning in the Data Mining era: Representations, generalization, and run-time.* PhD thesis, Ramon Llull University, Barcelona, Catalonia, Spain, 2004.

[9] J. Bacardit, E.K. Burke, and N. Krasnogor. Improving the scalability of rule-based evolutionary learning. *Memetic Computing*, 1(1):55–67, 2009.

[10] J. Bacardit and N. Krasnogor. Smart crossover operator with multiple parents for a pittsburgh learning classifier system. In *Proceedings of the 2006 Genetic and Evolutionary Computation Conference (GECCO)*, pages 1441–1448. ACM Press New York, NY, USA, July 2006. ISBN 1-59593-186-4.

[11] J. Bacardit, M. Stout, J.D. Hirst, J. Blazewicz, and N. Krasnogor. Coordination number prediction using learning classifier systems: performance and interpretability. In *Proceedings of the 2006 Genetic and Evolutionary Computation Conference (GECCO)*, pages 247–254. ACM Press New York, NY, USA, 2006. ISBN 1-59593-186-4.

[12] J. Bacardit, M. Stout, J.D. Hirst, K. Sastry, X. Llorà, and N. Krasnogor. Automated alphabet reduction method with evolutionary algorithms for protein structure prediction. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 346–353. ACM, 2007.

[13] J. Bacardit, M. Stout, J.D. Hirst, A. Valencia, R.E. Smith, and N. Krasnogor. Automated alphabet reduction for protein datasets. *BMC bioinformatics*, 10(1):6, 2009.

[14] I. Bakó, T. Megyes, S. Bálint, V. Chihaia, M.C. Bellissent-Funel, H. Krienke, A. Kopf, and S.H. Suh. Hydrogen bonded network properties in liquid formamide. *The Journal of chemical physics*, 132:014506, 2010.

[15] M. Bastian, S. Heymann, and M. Jacomy. Gephi: An open source software for exploring and manipulating networks, 2009.

[16] V. Batagelj and A. Mrvar. Some analyses of erdos collaboration graph. *Social Networks*, 22(2):173–186, 2000.

[17] J.K. Benedetti. On the nonparametric estimation of regression functions. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 248–253, 1977.

[18] K. Bhattacharya, G. Mukherjee, J. Saramaki, K. Kaski, and S.S. Manna. The International Trade Network: weighted network analysis and modelling. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(02):P02002+, March 2008.

[19] J. Blakes, O. Raz, U. Feige, J. Bacardit, P. Widera, T. Ben-Yehezkel, E. Shapiro, and N. Krasnogor. Heuristic for maximizing dna reuse in synthetic dna library assembly. *ACS synthetic biology*, 3(8):529–542, 2014.

[20] J. Blakes, O. Raz, N. Krasnogor, and E. Shapiro. A new algorithm for combinatorial dna library assembly. In M. Galdzicki, editor, *5th International Workshop on Bio-Design Automation (IWBDA 2013)*, 2013.

[21] M. Boguñá, R. Pastor-Satorras, A. Díaz-Guilera, and A. Arenas. Models of social networks based on social distance attachment. *Physical review E*, 70(5):056122, 2004.

[22] B. Bollobás. *Graph theory*. Elsevier, 1982.

[23] D. Bonchev and N. Trinajstić. Information theory, distance matrix, and molecular branching. *jcp*, 67:4517–4533, November 1977.

[24] U. Brandes. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25:163–177, 2001.

[25] U. Brandes. On variants of shortest-path betweenness centrality and their generic computation. *Social Networks*, 30(2):136–145, 2008.

[26] U. Brandes and C. Pich. Centrality estimation in large networks. *International Journal of Bifurcation and Chaos*, 17(07):2303–2318, 2007.

[27] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the Seventh International Conference on World Wide Web 7*, WWW7, pages 107–117, Amsterdam, The Netherlands, The Netherlands, 1998. Elsevier Science Publishers B. V.

[28] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the web. *Comput. Netw.*, 33:309–320, June 2000.

[29] D. Bu, Y. Zhao, L. Cai, H. Xue, X. Zhu, H. Lu, J. Zhang, S. Sun, L. Ling, N. Zhang, et al. Topological structure analysis of the protein–protein interaction network in budding yeast. *Nucleic acids research*, 31(9):2443–2450, 2003.

[30] E.K. Burke, M.R. Hyde, and G. Kendall. Evolving bin packing heuristics with genetic programming. In *PPSN*, pages 860–869, 2006.

[31] C. Butts. The complexity of social networks: theoretical and empirical findings. *Social Networks*, 23(1):31–72, 2001.

[32] C.T Butts. An axiomatic approach to network complexity. *Journal of Mathematical Sociology*, 24:273–301, 2000.

[33] D.S. Campo, Z. Dimitrova, R.J. Mitchell, J. Lara, and Y. Khudyakov. Coordinated evolution of the hepatitis C virus. *Proceedings of the National Academy of Sciences*, 105(28):9685–9690, July 2008.

[34] H. Cao, F.J. Romero-Campero, S. Heeb, M. Cámara, and N. Krasnogor. Evolving cell models for systems and synthetic biology. *Systems and synthetic biology*, 4(1):55–84, 2010.

[35] L.C. Carpi, P.M. Saco, O.A. Rosso, and M.G. Ravetti. Structural evolution of the tropical pacific climate network. *The European Physical Journal B-Condensed Matter and Complex Systems*, 85(11):1–7, 2012.

[36] T. Chalumeau, L. da Fontoura Costa, O. Laligant, and F. Meriaudeau. Texture discrimination using hierarchical complex networks. In *Signal Processing for Image Enhancement and Multimedia Processing*, pages 95–102. Springer, 2008.

[37] H. Chang, B.-B. Su, Y.-P. Zhou, and D.-R. He. Assortativity and act degree distribution of some collaboration networks. *Physica A: Statistical Mechanics and its Applications*, 383(2):687–702, 2007.

[38] I. Chitoran, F. Pellegrino, and E. Marsico. *Approaches to phonological complexity*, volume 16. Walter de Gruyter, 2009.

[39] Y.-M. Choi and H.-J. Kim. A Directed Network of Greek and Roman Mythology. *ArXiv Physics e-prints*, June 2005.

[40] J.C. Claussen. Offdiagonal complexity: A computationally quick complexity measure for graphs and networks. *Physica A: Statistical Mechanics and its Applications*, 375(1):365 – 373, 2007.

[41] W.W. Cohen. Enron email dataset. http://www.cs.cmu.edu/ enron/. Accessed in 2009.

[42] V. Colizza, A. Flammini, M.A. Serrano, and A. Vespignani. Detecting rich-club ordering in complex networks. *Nat Phys*, 2(2):110–115, February 2006.

[43] V. Colizza, R. Pastor-Satorras, and A. Vespignani. Reaction–diffusion processes and metapopulation models in heterogeneous networks. *Nature Physics*, 3(4):276–282, 2007.

[44] I. Contreras, I. Arnaldo, N. Krasnogor, and J.I. Hidalgo. Blind optimisation problem instance classification via enhanced universal similarity metric. *Memetic Computing*, 6(4):263–276, 2014.

[45] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.

[46] Christophe Coupé, Egidio Marsico, François Pellegrino, et al. Complex patterns in phonological systems.

[47] T.M. Cover and P.E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13:21–27, 1967.

[48] P. Crucitti, V. Latora, and S. Porta. Centrality measures in spatial networks of urban streets. *Physical Review E*, 73(3):036125, 2006.

[49] L. da Fontoura Costa. Avalanches of activation and spikes in neuronal complex networks. 2008.

[50] L. da Fontoura Costa, F.A. Rodrigues, C.C. Hilgetag, and M. Kaiser. Beyond the average: Detecting global singular nodes from local features in complex networks. *EPL (Europhysics Letters)*, 87(1):18008, 2009.

[51] L. da Fontoura Costa, F.A. Rodrigues, G. Travieso, and P.R. Villas Boas. Characterization of complex networks: A survey of measurements. *ADVANCES IN PHYSICS*, 56:167, 2007.

[52] L. da Fontoura Costa, O. Sporns, L. Antiqueira, M.G.V. Nunes, and O.N. Oliveira Jr. Correlations between structure and dynamics in complex networks. 2006.

[53] L. da Fontoura Costa, M.A. Rodrigues Tognetti, and F.N. Silva. Concentric characterization and classification of complex network nodes: Application to an institutional collaboration network. *Physica A: Statistical Mechanics and its Applications*, 387(24):6201–6214, 2008.

[54] A. Davis, B.B. Gardner, and M.R. Gardner. Deep south chicago, 1941.

[55] M. Dehmer, N. Barbarini, K. Varmuza, and A. Graber. A large scale analysis of information-theoretic network complexity measures using chemical structures. *PLoS ONE*, 4(12):e8057, 12 2009.

[56] M. Dehmer, N. Barbarini, K. Varmuza, and A. Graber. A large scale analysis of information-theoretic network complexity measures using chemical structures. *PLoS One*, 4(12):e8057, 2009.

[57] M. Dehmer, S. Borgert, and F. Emmert-Streib. Network classes and graph complexity measures. In *Proceedings of the 2008 First International Conference on Complexity and Intelligence of the Artificial and Natural Complex Systems. Medical Applications of the Complex Systems. Biomedical Computing*, pages 77–84, Washington, DC, USA, 2008. IEEE Computer Society.

[58] M. Dehmer and F. Emmert-Streib. Structural information content of networks: Graph entropy based on local vertex functionals. *Computational Biology and Chemistry*, 32(2):131 – 138, 2008.

[59] M. Dehmer and A. Mowshowitz. A history of graph entropy measures. *Information Sciences*, 181(1):57 – 78, 2011.

[60] M. Dehmer, K. Varmuza, S. Borgert, and F. Emmert-Streib. On entropy-based molecular descriptors: statistical analysis of real and synthetic chemical structures. *Journal of chemical information and modeling*, 49(7):1655–1663, 2009.

[61] D. Densmore, T.H.-C. Hsiau, J. Kittleson T, W. DeLoache, C. Batten, and J.C. Anderson. Algorithms for automated dna assembly. *Nucleic acids research*, page gkq165, 2010.

[62] P.A.M. Dirac. *The principles of quantum mechanics*. Number 27. Oxford university press, 1981.

[63] I.A. Doherty, N.S. Padian, C. Marlow, and S.O. Aral. Determinants and consequences of sexual networks as they affect the spread of sexually transmitted infections. *Journal of Infectious Diseases*, 191(Supplement 1):S42–S54, 2005.

[64] J.E. Dubois, G. Carrier, and A. Panaye. Darc topological descriptors for pattern recognition in molecular database management systems and design. *Journal of chemical information and computer sciences*, 31(4):574–578, 1991.

[65] J. Duch and A. Arenas. Community identification using extremal optimization phys. *Rev. E*, 72:027104, 2005.

[66] N. Eagle and A. Pentland. Reality mining: sensing complex social systems. *Personal and Ubiquitous Computing*, 10(4):255–268, 2006.

[67] C. Echtermeyer, L. da Fontoura Costa, F.A. Rodrigues, and M. Kaiser. Automatic network fingerprinting through single-node motifs. *PloS one*, 6(1):e15765, 2011.

[68] F. Emmert-Streib, L. Chen, and J. Storey. Functional Annotation of Genes in Saccharomyces cerevisiae based on Joint Betweenness. *ArXiv e-prints*, September 2007.

[69] F. Emmert-Streib and M. Dehmer. Information theoretic measures of uhg graphs with low computational complexity. *Applied Mathematics and Computation*, 190(2):1783 – 1794, 2007.

[70] F. Emmert-Streib and M. Dehmer. Networks for systems biology: conceptual connection of data and function. *Systems Biology, IET*, 5(3):185–207, 2011.

[71] F. Emmert-Streib and M. Dehmer. Exploring statistical and population aspects of network complexity. *PloS one*, 7(5):e34523, 2012.

[72] F. Escolano, E.R. Hancock, and M.A. Lozano. Heat diffusion: Thermodynamic depth complexity of networks. *Physical Review E*, 85(3):036206, 2012.

[73] E. Estrada and J.A. Rodríguez-Velázquez. Spectral measures of bipartivity in complex networks. *Physical Review E*, 72(4):046105, 2005.

[74] G. Fagiolo, J. Reyes, and S. Schiavo. World-trade web: Topological properties, dynamics, and evolution. 79(3):036115, March 2009.

[75] I.J. Farkas, I. Dernyi, A-L. Barabsi, T. Vicsek, and P.P. Setany. Spectra of real-world graphs: Beyond the semicircle law. *Physical Review E*, 64.

[76] M. Fire, R. Puzis, and Y. Elovici. Link prediction in highly fractional data sets. *Handbook of Computational Approaches to Counterterrorism*, 2012.

[77] J.L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378, 1971.

[78] F. Fomin and D. Thilikos. Branchwidth of graphs. In *Encyclopedia of Algorithms*, pages 101–105. Springer, 2008.

[79] L.C. Freeman, S.P. Borgatti, and D.R. White. Centrality in valued graphs: A measure of betweenness based on network flow. 1991.

[80] A.S. Fukunaga. Automated discovery of composite sat variable-selection heuristics. In *Eighteenth national conference on Artificial intelligence*, pages 641–648, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.

[81] A.S. Fukunaga. Evolving local search heuristics for SAT using genetic programming. In *Genetic and Evolutionary Computation – GECCO-2004, Part II*, volume 3103 of *Lecture Notes in Computer Science*, pages 483–494, Seattle, WA, USA, 26-30 June 2004. Springer-Verlag.

[82] J. Fürnkranz. Separate-and-conquer rule learning. *Artificial Intelligence Review*, 13(1):3–54, 1999.

[83] Gephi. Gephi datasets. https://wiki.gephi.org/index.php?title=Datasets.

[84] M. Girvan and M.E.J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.

[85] E. Glaab, A. Baudot, N. Krasnogor, and A. Valencia. Topogsa. *Bioinformatics*, 26:1271–1272, May 2010.

[86] P. M. Gleiser. How to become a superhero. *Journal of Statistical Mechanics: Theory and Experiment*, 9:20, September 2007.

[87] P.M. Gleiser and L. Danon. Community structure in jazz. *Advances in Complex Systems*, 06(04):565–573, 2003.

[88] P.M. Gleiser and V.I. Spoormaker. Modelling hierarchical structure in functional brain networks. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 368(1933):5633–5644, 2010.

[89] K.-I. Goh, M.E. Cusick, D. Valle, B. Childs, M. Vidal, and A.-L. Barabási. The human disease network. *Proceedings of the National Academy of Sciences*, 104(21):8685–8690, 2007.

[90] K.-I. Goh, B. Kahng, and D. Kim. Spectra and eigenvectors of scale-free networks. *Physical Review E*, 64, 2001.

[91] J.L. Gross and J. Yellen. *Graph theory and its applications*. CRC press, 2005.

[92] R. Guimera, L. Danon, A. Diaz-Guilera, F. Giralt, and A. Arenas. Self-similar community structure in a network of human interactions. *Phys. Rev. E*, 68:065103, Dec 2003.

[93] K.L. Gwet. *Handbook of inter-rater reliability: The definitive guide to measuring the extent of agreement among raters*. Advanced Analytics, LLC, 2014.

[94] P. Hage. Eccentricity and centrality in networks. *Social Networks*, 17(1):57–63, January 1995.

[95] Hamsterster. Hamsterster social network. http://www.hamsterster.com.

[96] W.E. Hart, N. Krasnogor, and J.E. Smith, editors. *Recent advances in memetic algorithms*, volume 166 of *Studies in Fuzzyness and Soft Computing*. Springer Berlin Heidelberg New York, 2004.

[97] J.H. Holland. Adaptation. *Progress in theoretical biology*, 4:263–293, 1976.

[98] J.H. Holland and J.S. Reitman. Cognitive systems based on adaptive algorithms. In *Pattern-directed Inference Systems*. New York: Academic Press, 1978.

[99] H.J. Gold J et al. *Mathematical modeling of biological systems. An introductory guidebook*. John Wiley & Sons, Inc., 1977.

[100] M. Jacomy, T. Venturini, S. Heymann, and M. Bastian. Forceatlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software. *PloS one*, 9(6):e98679, 2014.

[101] H. Jeong, S.P. Mason, A.-L. Barabasi, and Z.N. Oltvai. Lethality and centrality in protein networks. *arXiv preprint cond-mat/0105306*, 2001.

[102] D. Johnson. Finding all the elementary circuits of a directed graph. *SIAM Journal on Computing*, 4(1):77–84, 1975.

[103] K.A. De Jong and W.M. Spears. Learning concept classification rules using genetic algorithms. In *Proceedings of the Twelfth International Conference on Artificial Intelligence IJCAI-91*, volume 2, 1991.

[104] K. Jongkwang and W. Thomas. What is a complex graph? *Physica A: Statistical Mechanics and its Applications*, 387(11):2637 – 2652, 2008.

[105] M. Kaiser and C.C. Hilgetag. Edge vulnerability in neural and metabolic networks. *Biol. Cybern.*, 90:311–317, May 2004.

[106] F. Kaspar and H.G. Schuster. Easily calculable measure for the complexity of spatiotemporal patterns. *Physical Review A*, 36(2):842, 1987.

[107] H.-J. Kim and J. M. Kim. Cyclic topology in complex networks. 72(3):036109, September 2005.

[108] I.Z. Kiss, D.M. Green, and R.R. Kao. The network of sheep movements within great britain: network properties and their implications for infectious disease spread. *Journal of the Royal Society Interface*, 3(10):669–677, 2006.

[109] D.E. Knuth. *The Stanford GraphBase: a platform for combinatorial computing*, volume 37. Addison-Wesley Reading, 1993.

[110] N. Krasnogor. Self-generating metaheuristics in bioinformatics: the protein structure comparison case. *Genetic Programming and Evolvable Machines*, 5(2):181–201, 2004.

[111] N. Krasnogor and S. Gustafson. A study on the use of "self-generation" in memetic algorithms. *Natural Computing*, 3(1):53–76, 2004.

[112] N. Krasnogor and J.E. Smith. A tutorial for competent memetic algorithms: model, taxonomy and design issues. *IEEE Transactions on Evolutionary Computation*, 9(5):474–488, 2005.

[113] A. Lempel and J. Ziv. On the complexity of finite sequences. *Information Theory, IEEE Transactions on*, 22(1):75–81, 1976.

[114] J. Leskovec, D. Huttenlocher, and J. Kleinberg. Signed networks in social media. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1361–1370. ACM, 2010.

[115] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Trans. Knowl. Discov. Data*, 1(1), March 2007.

[116] J. Leskovec, K.J. Lang, A. Dasgupta, and M.W. Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics*, 6(1):29–123, 2009.

[117] K.S. Lew, T.S. Dillon, and K.E. Forward. Software complexity and its impact on software reliability. *Software Engineering, IEEE Transactions on*, 14(11):1645–1655, 1988.

[118] L. Li, J.M. Garibaldi, and N. Krasnogor. Automated self-assembly programming paradigm: The impact of network topology. *Int. J. Intell. Syst.*, 24:793–817, July 2009.

[119] Y. Liu, J-J. Slotine, and A-L. Barabasi. Controllability of complex networks. *Nature*, 473(7346):167–173, May 2011.

[120] D. Lusseau, L. Barrett, and S. P. Henzi. Formalising the multidimensional nature of social networks. *ArXiv e-prints*, January 2011.

[121] D. Lusseau and M.E.J. Newman. Identifying the role that animals play in their social networks. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 271(Suppl 6):S477–S481, 2004.

[122] P. Mahadevan, D. Krioukov, M. Fomenkov, X. Dimitropoulos, A. Vahdat, et al. The internet as-level topology: three data sources and one definitive metric. *SIGCOMM*, 36(1):17–26, 2006.

[123] B. Malko, D. Balcan, and A. Erzan. Information content based model for the topological properties of the gene regulatory network of escherichia coli. *Journal of Theoretical Biology*, 263(3):281 – 294, 2010.

[124] S. Maniu, T. Abdessalem, and B. Cautis. Casting a web of trust over wikipedia: an interaction-based approach. In *Proceedings of the 20th international conference companion on World wide web*, pages 87–88. ACM, 2011.

[125] O. Mason and M. Verwoerd. Graph theory and networks in biology. In *IET Systems Biology, 1:89 119*, pages 89–119, 2007.

[126] P. Massa and P. Avesani. Controversial users demand local trust metrics: An experimental study on epinions.com community. In *AAAI*, pages 121–126, 2005.

[127] P. Massa, M. Salvetti, and D. Tomasoni. Bowling alone and trust decline in social network sites. In *Dependable, Autonomic and Secure Computing, 2009. DASC'09. Eighth IEEE International Conference on*, pages 658–663. IEEE, 2009.

[128] R. Matei, A. Iamnitchi, and P. Foster. Mapping the gnutella network. *Internet Computing*, 6(1):50–57, 2002.

[129] P. Mateti and N. Deo. On Algorithms for Enumerating All Circuits of a Graph. *Siam Journal on Computing*, 5:90–99, 1976.

[130] B.W. Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2):442–451, 1975.

[131] J. J. McAuley, L. da Fontoura Costa, and T. S. Caetano. Rich-club phenomenon across complex network hierarchies. *Applied Physics Letters*, 91(8):084103, August 2007.

[132] R. Michalski, S. Palus, and P. Kazienko. Matching organizational structure and social network extracted from email communication. In *Lecture Notes in Business Information Processing*, volume 87, pages 197–206. Springer Berlin Heidelberg, 2011.

[133] A. Mislove, M. Marcon, K.P. Gummadi, P. Druschel, and B. Bhattacharjee. Measurement and analysis of online social networks. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 29–42. ACM, 2007.

[134] R. J. Mondragón and S. Zhou. Random networks with given rich-club coefficient. *European Physical Journal B*, 85:328, September 2012.

[135] Y. Moreno, J.B. Gómez, and A.F. Pacheco. Epidemic incidence in correlated complex networks. *Physical Review E*, 68(3):035103, 2003.

[136] J.-B. Mouret and J. Clune. An algorithm to create phenotype-fitness maps.

[137] J.-B. Mouret and J. Clune. Uncovering phenotype-fitness maps using mole. *connections*, 5:10.

[138] A. Mowshowitz and V. Mitsou. *Entropy, Orbits, and Spectra of Graphs*, pages 1–22. Wiley-VCH Verlag GmbH & Co. KGaA, 2009.

[139] F. Mufalli. *Complexity measures and congestion issues in wireless sensor networks*. ProQuest, 2008.

[140] N. Müllner. *Simulation of Self-Stabilizing Distributed Algorithms to Determine Fault Tolerance Measures*. PhD thesis, 2007.

[141] J.L. Myers, A. Well, and R.F. Lorch. *Research design and statistical analysis*. Routledge, 2010.

[142] A.A. Nanavati, S. Gurumurthy, G. Das, D. Chakraborty, K. Dasgupta, S. Mukherjea, and A. Joshi. On the structural properties of massive telecom call graphs: findings and implications. In *Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 435–444. ACM, 2006.

[143] D.L. Neel and M.E. Orrison. The linear complexity of a graph. *the electronic journal of combinatorics*, 13(R9):1, 2006.

[144] M.E.J. Newman. The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences*, 98(2):404–409, 2001.

[145] M.E.J. Newman. Mixing patterns in networks. *Physical Review E*, 67(2):026126, 2003.

[146] M.E.J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev. E*, 74:036104, Sep 2006.

[147] M.E.J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, 2006.

[148] R.E. Niemeyer. From ions to intrinsic connectivity networks: The neuroscience that bio-sociology needs to know. 2011.

[149] W. De Nooy, A. Mrvar, and V. Batagelj. *Exploratory social network analysis with Pajek*, volume 27. Cambridge University Press, 2011.

[150] N. Le Novère, B. Bornstein, A. Broicher, M. Courtot, M. Donizelli, H. Dharuri, L. Li, H. Sauro, M. Schilstra, B. Shapiro, J.L. Snoep, and M. Hucka. BioModels Database: a free, centralized database of curated, published, quantitative kinetic models of biochemical and cellular systems. *Nucleic Acids Research*, 34(Database issue):D689–D691, Jan 2006.

[151] T. Opsahl. Why anchorage is not (that) important: Binary ties and sample selection. 2011.

[152] T. Opsahl and P. Panzarasa. Clustering in weighted networks. *Social networks*, 31(2):155–163, 2009.

[153] S. Oum and P. Seymour. Approximating clique-width and branch-width. *Journal of Combinatorial Theory, Series B*, 96(4):514–528, 2006.

[154] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. 1999.

[155] R.K. Pan, N. Chatterjee, and S. Sinha. Mesoscopic organization reveals the constraints governing caenorhabditis elegans nervous system. *PLoS ONE*, 5(2):e9240, 02 2010.

[156] M. Parter, N. Kashtan, and U. Alon. Environmental variability and modularity of bacterial metabolic networks. *BMC Evolutionary Biology*, 7(1):169+, September 2007.

[157] K. Pearson. Note on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of London*, 58:240–242, 1895.

[158] Z. Qing-Bai, L. Meng-Jie, and C. Qi-Cai. Emergence of small-world and limitations to its maximization in a macaque cerebral cortical network. *Chinese Physics Letters*, 28(6):068703, 2011.

[159] R. Rabinovich, Der Rheinisch westflischen Technischen Hochschule Aachen, and Mathematische Grundlagen Der Informatik. Complexity measures for directed graphs, 2008.

[160] K.E. Read. Cultures of the central highlands, new guinea. *Southwestern Journal of Anthropology*, pages 1–43, 1954.

[161] J. Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978.

[162] N. Robertson and P.D. Seymour. Graph minors. x. obstructions to tree-decomposition. *Journal of Combinatorial Theory, Series B*, 52(2):153 – 190, 1991.

[163] F.J. Romero-Campero, H. Cao, M. Camara, and N. Krasnogor. Structure and parameter estimation for cell systems biology models. In *Proceedings of the 10th annual conference on Genetic and evolutionary computation*, pages 331–338. ACM, 2008.

[164] R.A. Rossi and N.K. Ahmed. The network data repository with interactive graph analytics and visualization. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[165] R.A. Rossi, D.F. Gleich, A.H. Gebremedhin, and M.A. Patwary. What if clique were fast? maximum cliques in information networks and strong components in temporal networks. *arXiv preprint arXiv:1210.5802*, pages 1–11, 2012.

[166] R.A. Rossi, D.F. Gleich, A.H. Gebremedhin, and M.A. Patwary. Fast maximum clique algorithms for large graphs. In *Proceedings of the 23rd International Conference on World Wide Web (WWW)*, 2014.

[167] M. Rosvall and C.T. Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105(4):1118–1123, 2008.

[168] M. Samec, A. Santiago, J.P. Cárdenas, R.M. Benito, A.M. Tarquis, S.J. Mooney, and D. Korošak. Quantifying soil complexity using network models of soil porous structure. *Nonlinear Processes in Geophysics*, 20:41–45, 2013.

[169] D. Sanassy, P. Widera, and N. Krasnogor. Meta-stochastic simulation of biochemical models for systems and synthetic biology. *ACS synthetic biology*, 2014.

[170] S. Scellato, A. Cardillo, V. Latora, and S. Porta. The backbone of a city. *The European Physical Journal B-Condensed Matter and Complex Systems*, 50(1):221–225, 2006.

[171] O. Schmitt, P. Eipert, K. Philipp, R. Kettlitz, G. Fuellen, and A. Wree. The intrinsic connectome of the rat amygdala. *Frontiers in Neural Circuits*, 6(81), 2012.

[172] B. Scholkopf, A.J. Smola, R. Williamson, and P.L. Bartlett. New support vector algorithms. *Neural Computation*, 12:1207–1245, 2000.

[173] C.E. Shannon. Prediction and Entropy of Printed English. *The Bell System Technical Journal*, 30:50–64, 1951.

[174] F. Shi, C. Li, D. Qin, Y. Zhu, and F. Yang. A complexity measure for military communication networks. In *MILITARY COMMUNICATIONS CONFERENCE, 2011-MILCOM 2011*, pages 1708–1713. IEEE, 2011.

[175] P. Siepmann, C.P. Martin, I. Vancea, P.J. Moriarty, and N. Krasnogor. A genetic algorithm approach to probing the evolution of self-organized nanostructured systems. *Nano letters*, 7(7):1985–1990, 2007.

[176] A. Sih, S.F. Hanser, and K.A. McHugh. Social network theory: new insights and issues for behavioral ecologists. *Behavioral Ecology and Sociobiology*, 63(7):975–988, 2009.

[177] F.N. Silva. Hierarchical characterization of complex networks. *Journal of Statistical Physics*, 125:841–872, 2006.

[178] R. Singh, J. Xu, and B. Berger. Global alignment of multiple protein interaction networks with application to functional orthology detection. *PNAS*, 105(35):12763–12768, 2008.

[179] J. Siriwardana and S.K. Halgamuge. Fast shortest path optimization inspired by shuttle streaming of physarum polycephalum. In *Evolutionary Computation (CEC), 2012 IEEE Congress on*, pages 1–8. IEEE, 2012.

[180] J.E. Smith. Co-evolving memetic algorithms: A review and progress report. *IEEE Transactions in Systems, Man and Cybernetics, part B*, 37(1):6–17, 2007.

[181] SocioPatterns. Infectious contact networks. http://www.sociopatterns.org/datasets/. Accessed 09/12/12.

[182] R.V. Sol. Information Theory of Complex Networks : on evolution and architectural constraints. *Most*, pages 1–15.

[183] J. Song, K. Takemoto, H. Shen, H. Tan, M.M. Gromiha, and T. Akutsu. Prediction of Protein Folding Rates from Structural Topology and Complex Network Properties. *IPSJ Transactions on Bioinformatics*, 3:40–53, 2010.

[184] N. Spring, R. Mahajan, and D. Wetherall. Measuring ISP topologies with rocketfuel. In *SIGCOMM*, volume 32, pages 133–145, 2002.

[185] M.I. Stankevich, I.V. Stankevich, and N.S. Zefirov. Topological indices in organic chemistry. *Russian Chemical Reviews*, 57(3):191, 1988.

[186] M. Stout, J. Bacardit, J.D. Hirst, J. Blacewicz, and N. Krasnogor. Prediction of residue exposure and contact number for simplified hp lattice model proteins using learning classifier systems. In *Proceedings of the 7th International FLINS Conference on Applied Artificial Intelligence*, pages 601–608. World Scientific, August 2006.

[187] M. Stout, J. Bacardit, J.D. Hirst, and N. Krasnogor. Prediction of recursive convex hull class assignments for protein residues. *Bioinformatics*, 24(7):916–923, 2008.

[188] M. Stout, J. Bacardit, J.D. Hirst, N. Krasnogor, and J. Blazewicz. From hp lattice models to real proteins: coordination number prediction using learning classifier systems. In *4th European Workshop on Evolutionary Computation and Machine Learning in Bioinformatics*, volume 3907 of *Springer Lecture Notes in Computer Science*, pages 208–220, Budapest, Hungary, April 2006. Springer. ISBN 978-3-540-33237-4.

[189] M. Stout, J. Bacardit, J.D. Hirst, R.E. Smith, and N. Krasnogor. Prediction of topological contacts in proteins using learning classifier systems. *Soft Computing, Special Issue on Evolutionary and Metaheuristic-based Data Mining (EMBDM), (in press)*, 2007.

[190] M. Tabacman. Discovering the limits of machine learning: use of lcs to predict tsp structure. Master's thesis, Facultad de Ciencias Exactas y Naturales. Universidad de Buenos Aires, 2008.

[191] K. Takemoto and T. Akutsu. Origin of structural difference in metabolic networks with respect to temperature. *BMC Systems Biology*, 2(1):82+, 2008.

[192] T. Tamura, N. Christian, K. Takemoto, O. Ebenhöh, and T. Akutsu. Analysis and prediction of nutritional requirements using structural properties of metabolic networks and support vector machines. *Genome Informatics Series*, 22:176–190, 2009.

[193] G. Terrazas, P. Siepmann, G. Kendall, and N. Krasnogor. An evolutionary methodology for the automated design of cellular automaton-based complex systems. *J. Cellular Automata*, 2(1):77–102, 2007.

[194] R. Todeschini and V. Consonni. Handbook of molecular descriptors. 2000.

[195] Truthy. Information diffusion research at indiana university. http://truthy.indiana.edu/. Accessed 10/20/12.

[196] J.W. Tukey. Exploratory data analysis. *Reading, Ma*, 231:43–44, 1977.

[197] J. Wang, P. Neskovic, and L.N. Cooper. Improving nearest neighbor rule with a simple adaptative distance measure. *Pattern Recognition Letters*, 28:207–213, 2007.

[198] X. Wang, S. Guan, and C.H. Lai. Protecting infrastructure networks from cost-based attacks. *New Journal of Physics*, 11(3):033006, 2009.

[199] D.J. Watts and S.H. Strogatz. Collective dynamics of small-worldnetworks. *nature*, 393(6684):440–442, 1998.

[200] D. Wildenschild and A.P. Sheppard. X-ray imaging and analysis techniques for quantifying pore-scale structure and processes in subsurface porous medium systems. *Advances in Water Resources*, 2012.

[201] T. Wilhelm and J. Hollunder. Information theoretic description of networks. *Physica A: Statistical Mechanics and its Applications*, 385(1):385 – 396, 2007.

[202] J. Winerip. Graph linear complexity. 2008.

[203] D.H. Wolpert and W.G. Macready. No free lunch theorems for search. Technical Report SFI-TR-95-02-010, Santa Fe, NM, 1995.

[204] D.H. Wolpert and W.G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, April 1997.

[205] Y. Wu, K. Ianakiev, and V. Govindaraju. Improved k-nearest neighbor classification. *Pattern Recognition*, 35(10):2311–2318, 2002.

[206] Y. Wu, L. Sun, and H. Liu. Property description and comparison of blog community at different scales. In *Computer Research and Development (ICCRD), 2011 3rd International Conference on*, volume 2, pages 257 –261, march 2011.

[207] C. Xie, S. Guo, R. Rejaie, and Y. Pan. Examining graph properties of unstructured peer-to-peer overlay topology. In *IEEE Global Internet Symposium, 2007*, pages 13 –18, may 2007.

[208] X. Xueying and L. Xin. A growing model for biological networks on recombination and divergence principle. In *Bioinformatics and Biomedical Engineering, (iCBBE) 2011 5th International Conference on*, pages 1 –4, may 2011.

[209] W.W. Zachary. An information flow model for conflict and fission in small groups. *Journal of anthropological research*, pages 452–473, 1977.

[210] X. Zhang, H. Zhao, L. Wang, and C. Li. On the internet as-level topology evolution basing on hierarchy characteristic. In *Control and Decision Conference, 2008. CCDC 2008. Chinese*, pages 4470 –4475, july 2008.

[211] Y. Zhang, Z. Li, B. Wu, and S. Wu. A spaceflight operation complexity measure and its experimental validation. *International Journal of Industrial Ergonomics*, 39(5):756–765, 2009.

[212] S. Zhou, I. Cox, and V. Petricek. Characterising web site link structure. In *Web Site Evolution, 2007. WSE 2007. 9th IEEE International Workshop on*, pages 73 –80, oct. 2007.

[213] S. Zhou and R.J. Mondragon. The rich-club phenomenon in the internet topology. *Communications Letters, IEEE*, 8(3):180 – 182, march 2004.

List of Figures

2.1 Undirected Graph 8
2.2 Directed Graph 8
2.3 Weighted Undirected Graph 9
2.4 Weighted Directed Graph 9
2.5 Connected Graph Example 10
2.6 Disconnected Graph Example 10
2.7 Shortest Path Between v_3 and v_7 in an unweighted graph 13
2.8 Average Shortest Path Lengths 14
2.9 Betweenness 14
2.10 Eccentricity 15
2.11 Clustering Coefficient in the Connected Graph Example 16
2.12 Comparison of networks with different values of Average Shortest Path Length. Built using Force Atlas 2 algorithm [100] with Gephi [15]. Nodes represent network vertices, those with higher degree are larger and present a darker shade. Zoom is the closest possible without leaving any node outside. Edges not shown for visualizations where they would block the node distribution. . . 23
2.13 Comparison of networks with different values of Betweenness Centrality. Node size and darker shade increases according to the observed value for the measure. 25
2.14 Comparison of networks with different values of Diameter 29
2.15 Comparison of networks with different values of Average Clustering Coefficient 30
2.16 Comparison of networks with different values of Offdiagonal Complexity 32
2.17 Comparison of networks with different values of Eigenvector Centrality 33

2.18 Comparison of networks with different values of PageRank . . . 34

3.1 Instance distribution where K Nearest Neighbors is a good option. There is a clear distinction between elements of class Diamond and elements of class Cross . . . 58
3.2 Instance distribution where K Nearest Neighbors is not a good option . . . 58
3.3 SVM components: optimal margin and optimal hyperplane . . . 60
3.4 Example of a soft margin in SVM . . . 60
3.5 Example of a higher degree polynomial used in a SVM . . . 61

4.1 Visualization of four different networks from those selected for the Assorted Biological domain . . . 72
4.2 Visualization of four different networks from those selected for the Code domain . . . 74
4.3 Visualization of four different networks from those selected for the Collaboration domain . . . 77
4.4 Visualization of four different networks from those selected for the Combinatorial DNA Library Synthesis domain . . . 79
4.5 Visualization of four different networks from those selected for the Communication domain . . . 82
4.6 Visualization of four different networks from those selected for the Infrastructure domain . . . 84
4.7 Visualization of four different networks from those selected for the Metabolic domain . . . 86
4.8 Visualization of four different networks from those selected for the Reaction Dependency Graph domain . . . 88
4.9 Visualization of four different networks from those selected for the Social domain . . . 91
4.10 Visualization of four different networks from those selected for the Species Network Graph domain . . . 93
4.11 Visualization of four different networks from those selected for the Tweet domain . . . 95
4.12 Correlation between the studied measures for all networks in the Assorted Biological domain . . . 98
4.13 Correlation between the studied measures for all networks in the Code domain . . . 98
4.14 Correlation between the studied measures for all networks in the Collaboration domain . . . 99

4.15 Correlation between the studied measures for all networks in the Combinatorial DNA Library Synthesis domain 99
4.16 Correlation between the studied measures for all networks in the Communication domain 99
4.17 Correlation between the studied measures for all networks in the Infrastructure domain 100
4.18 Correlation between the studied measures for all networks in the Metabolic domain 100
4.19 Correlation between the studied measures for all networks in the Reaction Dependency Graph domain 101
4.20 Correlation between the studied measures for all networks in the Social domain 101
4.21 Correlation between the studied measures for all networks in the Species Network Graph domain 102
4.22 Correlation between the studied measures for all networks in the Tweet domain 102
4.23 Correlation between the studied measures for all networks in all domains 103
4.24 Statistical distribution of the observed values for $|V|$.The values computed for all networks have been normalized between 0 and 1. The boxes span from the first to the third quartile of each data series, with an inner line representing the median. Lower and upper whiskers represent minimum and maximum observed values respectively. 105
4.25 Statistical distribution of the observed values for $|E|$ 106
4.26 Statistical distribution of the observed values for Maximum Degree 106
4.27 Statistical distribution of the observed values for Diameter . . . 107
4.28 Statistical distribution of the observed values for PathLength . . 107
4.29 Statistical distribution of the observed values for Codelength . . 108
4.30 Statistical distribution of the observed values for Offdiagonal Complexity 109
4.31 Statistical distribution of the observed values for Global Efficiency 109
4.32 Statistical distribution of the observed values for average Betweenness Centrality 110
4.33 Statistical distribution of the observed values for minimum Betweenness Centrality 111
4.34 Statistical distribution of the observed values for maximum Betweenness Centrality 111

4.35 Statistical distribution of the observed values for median Betweenness Centrality . . . 112
4.36 Statistical distribution of the observed values for first quartile Betweenness Centrality . . . 113
4.37 Statistical distribution of the observed values for third quartile Betweenness Centrality . . . 113
4.38 Statistical distribution of the observed values for inter-quartile range Betweenness Centrality . . . 114
4.39 Statistical distribution of the observed values for average Closeness Centrality . . . 115
4.40 Statistical distribution of the observed values for minimum Closeness Centrality . . . 115
4.41 Statistical distribution of the observed values for maximum Closeness Centrality . . . 116
4.42 Statistical distribution of the observed values for median Closeness Centrality . . . 116
4.43 Statistical distribution of the observed values for first quartile Closeness Centrality . . . 117
4.44 Statistical distribution of the observed values for third quartile Closeness Centrality . . . 117
4.45 Statistical distribution of the observed values for inter-quartile range Closeness Centrality . . . 118
4.46 Statistical distribution of the observed values for average Clustering Coefficient . . . 119
4.47 Statistical distribution of the observed values for minimum Clustering Coefficient . . . 119
4.48 Statistical distribution of the observed values for maximum Clustering Coefficient . . . 120
4.49 Statistical distribution of the observed values for median Clustering Coefficient . . . 121
4.50 Statistical distribution of the observed values for first quartile Clustering Coefficient . . . 121
4.51 Statistical distribution of the observed values for third quartile Clustering Coefficient . . . 122
4.52 Statistical distribution of the observed values for inter-quartile range Clustering Coefficient . . . 122
4.53 Statistical distribution of the observed values for average Eigenvector Centrality . . . 123

4.54 Statistical distribution of the observed values for minimum Eigenvector Centrality . . . 123
4.55 Statistical distribution of the observed values for maximum Eigenvector Centrality . . . 124
4.56 Statistical distribution of the observed values for median Eigenvector Centrality . . . 124
4.57 Statistical distribution of the observed values for first quartile Eigenvector Centrality . . . 125
4.58 Statistical distribution of the observed values for third quartile Eigenvector Centrality . . . 125
4.59 Statistical distribution of the observed values for inter-quartile range Eigenvector Centrality . . . 126
4.60 Statistical distribution of the observed values for average PageRank 126
4.61 Statistical distribution of the observed values for minimum PageRank . . . 127
4.62 Statistical distribution of the observed values for maximum PageRank . . . 127
4.63 Statistical distribution of the observed values for median PageRank 128
4.64 Statistical distribution of the observed values for first quartile PageRank . . . 128
4.65 Statistical distribution of the observed values for third quartile PageRank . . . 129
4.66 Statistical distribution of the observed values for inter-quartile range PageRank . . . 129

5.1 Examples of CiGRAM networks with different number of nodes 134
5.2 Examples of CiGRAM networks with different density . . . 135
5.3 Examples of CiGRAM networks with different number of communities . . . 135
5.4 Examples of CiGRAM networks with different assortativity . . . 135
5.5 Correlation between the selected measures for the generated networks . . . 137
5.6 Statistical distribution of the observed values for Diameter in the artificial networks generated . . . 137
5.7 Statistical distribution of the observed values for PathLength in the artificial networks generated . . . 138
5.8 Statistical distribution of the observed values for Codelength in the artificial networks generated . . . 139

5.9 Statistical distribution of the observed values for Offdiagonal Complexity in the artificial networks generated 139
5.10 Statistical distribution of the observed values for Global Efficiency in the artificial networks generated 140
5.11 Statistical distribution of the observed values for average Betweenness Centrality in the artificial networks generated 141
5.12 Statistical distribution of the observed values for average Closeness Centrality in the artificial networks generated 141
5.13 Statistical distribution of the observed values for average Clustering Coefficient in the artificial networks generated 142
5.14 Statistical distribution of the observed values for average Eigenvector Centrality in the artificial networks generated 142
5.15 Statistical distribution of the observed values for average PageRank in the artificial networks generated 143
5.16 Histogram for KNN-C testing accuracies of CiGRAM networks. A uniform result is produced due to the lack of pseudo-random numbers in the algorithm. 147
5.17 Histogram for IKNN-C testing accuracies of CiGRAM networks. Since all runs provided the same result, the observed values present a uniform distribution. 148
5.18 Histogram for C_SVM-C testing accuracies of CiGRAM networks 149
5.19 Histogram for NU_SVM-C testing accuracies of CiGRAM networks 150
5.20 Histogram for BioHEL-C testing accuracies of CiGRAM networks 152

6.1 Histogram for BioHEL-C testing accuracies when considering all real networks reviewed, using Keel automatic 10-fold 159
6.2 Correlation between the selected measures when the outliers are removed 162
6.3 Histogram for BioHEL-C testing accuracies when discarding outliers, using Keel automatic 10-fold 163
6.4 Histogram for BioHEL-C testing accuracies when discarding outliers, using 70% / 30% manual partitioning based on $|V|$ 164
6.5 Histogram for BioHEL-C testing accuracies when discarding outliers, using 70% / 30% manual partitioning based on $|E|$ 166
6.6 Histogram for BioHEL-C testing accuracies when discarding outliers, using 90% / 10% manual partitioning based on $|V|$ 167
6.7 Histogram for BioHEL-C testing accuracies when discarding outliers, using 90% / 10% manual partitioning based on $|E|$ 168

List of Tables

2.1 Undirected Unweighed Graph Adjacency Matrix 10
2.2 Directed Unweighed Graph Adjacency Matrix 11
2.3 Undirected Weighed Graph Adjacency Matrix 11
2.4 Directed Weighed Graph Adjacency Matrix 11

4.1 Assorted Biological Networks 70
4.2 Code Networks 73
4.3 Collaboration Networks 75
4.4 Combinatorial DNA Library Synthesis Networks 78
4.5 Communication Networks 80
4.6 Infrastructure Networks 83
4.7 Metabolic Networks 85
4.8 Reaction Dependency Graph Networks 87
4.9 Social Networks 89
4.10 Species Network Graph Networks 92
4.11 Tweet Networks 94
4.12 Abbreviatons for the Complexity Measures in the Correlation results 97
4.13 Relative Running Time 104

5.1 Artificial Networks to be created with CiGRAM 136
5.2 Keel algorithms to classify our CiGRAM networks 145
5.3 Parameters used for the Keel algorithms 145
5.4 Simplified names for the artificial classes 146
5.5 Training Set Confusion Matrix for KNN-C for CiGRAM networks 146
5.6 Testing Set Confusion Matrix for KNN-C for CiGRAM networks 146
5.7 Training Set Confusion Matrix for IKNN-C for CiGRAM networks 147
5.8 Testing Set Confusion Matrix for IKNN-C for CiGRAM networks 148

5.9 Training Set Confusion Matrix for C_SVM-C for CiGRAM networks . . . 148
5.10 Testing Set Confusion Matrix for C_SVM-C for CiGRAM networks 149
5.11 Training Set Confusion Matrix for NU_SVM-C for CiGRAM networks . . . 150
5.12 Testing Set Confusion Matrix for NU_SVM-C for CiGRAM networks . . . 150
5.13 Training Set Confusion Matrix for BioHEL-C for CiGRAM networks . . . 151
5.14 Testing Set Confusion Matrix for BioHEL-C for CiGRAM networks 151
5.15 Classification Accuracy for the Artificial Networks . . . 153
5.16 Rules produced by BioHEL for the Artificial Networks . . . 153

6.1 Classification accuracy when considering all real networks reviewed 158
6.2 Number of rules and number of attributes per rule, produced by BioHEL in the artificial and real networks experiments. . . . 160
6.3 Outlier instances . . . 161
6.4 Classification accuracy when not considering outliers (automatic partitioning) . . . 162
6.5 Number of rules and number of attributes per rule, produced by BioHEL when discarding outliers, using Keel automatic 10-fold . 163
6.6 No Outliers Classification Accuracy (70% / 30% manual partitioning based on $|V|$) . . . 164
6.7 Number of rules and number of attributes per rule, produced by BioHEL when discarding outliers, using 70% / 30% manual partitioning based on $|V|$ . . . 165
6.8 No Outliers Classification Accuracy (70% / 30% manual partitioning based on $|E|$) . . . 165
6.9 Number of rules and number of attributes per rule, produced by BioHEL when discarding outliers, using 70% / 30% manual partitioning based on $|E|$ . . . 166
6.10 No Outliers Classification Accuracy (90% / 10% manual partitioning based on $|V|$) . . . 166
6.11 Number of rules and number of attributes per rule, produced by BioHEL when discarding outliers, using 90% / 10% manual partitioning based on $|V|$ . . . 167
6.12 No Outliers Classification Accuracy (90% / 10% manual partitioning based on $|E|$) . . . 168

6.13 Best ruleset produced by BioHEL when discarding outliers, using 90% / 10% manual partitioning based on $|E|$ 169
6.14 Number of rules and number of attributes per rule, produced by BioHEL when discarding outliers, using 90% / 10% manual partitioning based on $|E|$ 170
6.15 Classification Accuracy for Outliers as testing set 170

A.1 Manually selected 30% based on $|V|$ 186
A.2 Manually selected 10% based on $|V|$ 187
A.3 Manually selected 30% based on $|E|$ 188
A.4 Manually selected 10% based on $|E|$ 189
A.5 Abbreviations names for the real network domains in the experimental result tables 190
A.6 All Instances (Automatic Partitioning) KNN-C Training Set Confusion Matrix 190
A.7 All Instances (Automatic Partitioning) KNN-C Testing Set Confusion Matrix 191
A.8 All Instances (Automatic Partitioning) IKNN-C Training Set Confusion Matrix 191
A.9 All Instances (Automatic Partitioning) IKNN-C Testing Set Confusion Matrix 191
A.10 All Instances (Automatic Partitioning) SVM-C Training Set Confusion Matrix 192
A.11 All Instances (Automatic Partitioning) SVM-C Testing Set Confusion Matrix 192
A.12 All Instances (Automatic Partitioning) NUSVM-C Training Set Confusion Matrix 192
A.13 All Instances (Automatic Partitioning) NUSVM-C Testing Set Confusion Matrix 193
A.14 All Instances (Automatic Partitioning) BioHEL-C Training Set Confusion Matrix 193
A.15 All Instances (Automatic Partitioning) BioHEL-C Testing Set Confusion Matrix 193
A.16 Non Outliers (Automatic Partitioning) KNN-C Training Set Confusion Matrix 194
A.17 Non Outliers (Automatic Partitioning) KNN-C Testing Set Confusion Matrix 194
A.18 Non Outliers (Automatic Partitioning) IKNN-C Training Set Confusion Matrix 194

A.19 Non Outliers (Automatic Partitioning) IKNN-C Testing Set Confusion Matrix . . . 195
A.20 Non Outliers (Automatic Partitioning) SVM-C Training Set Confusion Matrix . . . 195
A.21 Non Outliers (Automatic Partitioning) SVM-C Testing Set Confusion Matrix . . . 195
A.22 Non Outliers (Automatic Partitioning) NUSVM-C Training Set Confusion Matrix . . . 196
A.23 Non Outliers (Automatic Partitioning) NUSVM-C Testing Set Confusion Matrix . . . 196
A.24 Non Outliers (Automatic Partitioning) BioHEL-C Training Set Confusion Matrix . . . 196
A.25 Non Outliers (Automatic Partitioning) BioHEL-C Testing Set Confusion Matrix . . . 197
A.26 Non Outliers (70/30 Vertex based) KNN-C Training Set Confusion Matrix . . . 197
A.27 Non Outliers (70/30 Vertex based) KNN-C Testing Set Confusion Matrix . . . 197
A.28 Non Outliers (70/30 Vertex based) IKNN-C Training Set Confusion Matrix . . . 198
A.29 Non Outliers (70/30 Vertex based) IKNN-C Testing Set Confusion Matrix . . . 198
A.30 Non Outliers (70/30 Vertex based) SVM-C Training Set Confusion Matrix . . . 198
A.31 Non Outliers (70/30 Vertex based) SVM-C Testing Set Confusion Matrix . . . 199
A.32 Non Outliers (70/30 Vertex based) NUSVM-C Training Set Confusion Matrix . . . 199
A.33 Non Outliers (70/30 Vertex based) NUSVM-C Testing Set Confusion Matrix . . . 199
A.34 Non Outliers (70/30 Vertex based) BioHEL-C Training Set Confusion Matrix . . . 200
A.35 Non Outliers (70/30 Vertex based) BioHEL-C Testing Set Confusion Matrix . . . 200
A.36 Non Outliers (70/30 Edge based) KNN-C Training Set Confusion Matrix . . . 200
A.37 Non Outliers (70/30 Edge based) KNN-C Testing Set Confusion Matrix . . . 201

A.38 Non Outliers (70/30 Edge based) IKNN-C Training Set Confusion Matrix 201
A.39 Non Outliers (70/30 Edge based) IKNN-C Testing Set Confusion Matrix 201
A.40 Non Outliers (70/30 Edge based) SVM-C Training Set Confusion Matrix 202
A.41 Non Outliers (70/30 Edge based) SVM-C Testing Set Confusion Matrix 202
A.42 Non Outliers (70/30 Edge based) NUSVM-C Training Set Confusion Matrix 202
A.43 Non Outliers (70/30 Edge based) NUSVM-C Testing Set Confusion Matrix 203
A.44 Non Outliers (70/30 Edge based) BioHEL-C Training Set Confusion Matrix 203
A.45 Non Outliers (70/30 Edge based) BioHEL-C Testing Set Confusion Matrix 203
A.46 Non Outliers (90/10 Vertex based) KNN-C Training Set Confusion Matrix 204
A.47 Non Outliers (90/10 Vertex based) KNN-C Testing Set Confusion Matrix 204
A.48 Non Outliers (90/10 Vertex based) IKNN-C Training Set Confusion Matrix 204
A.49 Non Outliers (90/10 Vertex based) IKNN-C Testing Set Confusion Matrix 205
A.50 Non Outliers (90/10 Vertex based) SVM-C Training Set Confusion Matrix 205
A.51 Non Outliers (90/10 Vertex based) SVM-C Testing Set Confusion Matrix 205
A.52 Non Outliers (90/10 Vertex based) NUSVM-C Training Set Confusion Matrix 206
A.53 Non Outliers (90/10 Vertex based) NUSVM-C Testing Set Confusion Matrix 206
A.54 Non Outliers (90/10 Vertex based) BioHEL-C Training Set Confusion Matrix 206
A.55 Non Outliers (90/10 Vertex based) BioHEL-C Testing Set Confusion Matrix 207
A.56 Non Outliers (90/10 Edge based) KNN-C Training Set Confusion Matrix 207

A.57 Non Outliers (90/10 Edge based) KNN-C Testing Set Confusion Matrix . . . 207
A.58 Non Outliers (90/10 Edge based) IKNN-C Training Set Confusion Matrix . . . 208
A.59 Non Outliers (90/10 Edge based) IKNN-C Testing Set Confusion Matrix . . . 208
A.60 Non Outliers (90/10 Edge based) SVM-C Training Set Confusion Matrix . . . 208
A.61 Non Outliers (90/10 Edge based) SVM-C Testing Set Confusion Matrix . . . 209
A.62 Non Outliers (90/10 Edge based) NUSVM-C Training Set Confusion Matrix . . . 209
A.63 Non Outliers (90/10 Edge based) NUSVM-C Testing Set Confusion Matrix . . . 209
A.64 Non Outliers (90/10 Edge based) BioHEL-C Training Set Confusion Matrix . . . 210
A.65 Non Outliers (90/10 Edge based) BioHEL-C Testing Set Confusion Matrix . . . 210
A.66 Non Outliers Vs Outliers (Automatic Partitioning) KNN-C Testing Set Confusion Matrix . . . 210
A.67 Non Outliers Vs Outliers (Automatic Partitioning) IKNN-C Testing Set Confusion Matrix . . . 211
A.68 Non Outliers Vs Outliers (Automatic Partitioning) SVM-C Testing Set Confusion Matrix . . . 211
A.69 Non Outliers Vs Outliers (Automatic Partitioning) NUSVM-C Testing Set Confusion Matrix . . . 211
A.70 Non Outliers Vs Outliers (Automatic Partitioning) BioHEL-C Testing Set Confusion Matrix . . . 212
A.71 Vertex Based 70 vs Outliers KNN-C Testing Set Confusion Matrix 212
A.72 Vertex Based 70 vs Outliers IKNN-C Testing Set Confusion Matrix 212
A.73 Vertex Based 70 vs Outliers SVM-C Testing Set Confusion Matrix 213
A.74 Vertex Based 70 vs Outliers NUSVM-C Testing Set Confusion Matrix . . . 213
A.75 Vertex Based 70 vs Outliers BioHEL-C Testing Set Confusion Matrix . . . 213
A.76 Edge Based 70 vs Outliers KNN-C Testing Set Confusion Matrix 213
A.77 Edge Based 70 vs Outliers IKNN-C Testing Set Confusion Matrix 214
A.78 Edge Based 70 vs Outliers SVM-C Testing Set Confusion Matrix 214
A.79 Edge Based 70 vs Outliers NUSVM-C Testing Set Confusion Matrix 214

A.80 Edge Based 70 vs Outliers BioHEL-C Testing Set Confusion Matrix 214
A.81 Vertex Based 90 vs Outliers KNN-C Testing Set Confusion Matrix 215
A.82 Vertex Based 90 vs Outliers IKNN-C Testing Set Confusion Matrix 215
A.83 Vertex Based 90 vs Outliers SVM-C Testing Set Confusion Matrix 215
A.84 Vertex Based 90 vs Outliers NUSVM-C Testing Set Confusion Matrix 215
A.85 Vertex Based 90 vs Outliers BioHEL-C Testing Set Confusion Matrix 216
A.86 Edge Based 90 vs Outliers KNN-C Testing Set Confusion Matrix 216
A.87 Edge Based 90 vs Outliers IKNN-C Testing Set Confusion Matrix 216
A.88 Edge Based 90 vs Outliers SVM-C Testing Set Confusion Matrix 216
A.89 Edge Based 90 vs Outliers NUSVM-C Testing Set Confusion Matrix 217
A.90 Edge Based 90 vs Outliers BioHEL-C Testing Set Confusion Matrix 217

Index

Branch Decomposition, 48, 179

Capacity, 14
CiGRAM, 134, 179, 183
Concentric Clustering Coefficient, 17
Concentric Level, 17
Concentric Node Degree, 17
Concentric Ring, 17
Connected graph, 9

Default Classification, 63
Degree, 9

Evolutionary Computation, 62

Fleiss' Kappa, 179

GABIL, 64
GAssist, 65
Genetic Algorithm, 62
Graph, 8

Hierarchical Level, 17

K Nearest Neighbors, 57
Keel, 144

Learning Classifier Systems, 61

Machine Learning, 62
Maximum Flow, 14

Neighbor, 8
Neighborhood, 15

Path, 9
Pittsburgh approach, 62

Reaction Dependency Graph, 87
Reinforcement Learning, 64
Remaining Degree Distribution, 39
Rich Club, 21

Shortest Path Length, 13
Species Network Graph, 92
SPL, 13
Supervised Learning, 54
Support Vector Machines, 59

Topological Index, 22
Total System Throughflow, 44

UHG, 46
Universal Hierarchical Graph, 46
Unsupervised Learning, 55

Weighted graph, 9