Università degli Studi di Pisa

Dipartimento di Informatica
Dottorato di Ricerca in Informatica

Settore Scientifico Disciplinare: INF/01

Ph.D. Thesis

Social Network Dynamics

Giulio Rossetti

Supervisor
Dino Pedreschi
University of Pisa

Supervisor
Fosca Giannotti
ISTI-CNR

May 5, 2015

Abstract

This thesis focuses on the analysis of structural and topological network problems. In particular, in this work the privileged subjects of investigation will be both static and dynamic social networks. Nowadays, the constantly growing availability of *Big Data* describing human behaviors (i.e., the ones provided by online social networks, telco companies, insurances, airline companies...) offers the chance to evaluate and validate, on large scale realities, the performances of algorithmic approaches and the soundness of sociological theories. In this scenario, exploiting data-driven methodologies enables for a more careful modeling and thorough understanding of observed phenomena. In the last decade, graph theory has lived a second youth: the scientific community has extensively adopted, and sharpened, its tools to shape the so called *Network Science*. Within this highly active field of research, it is recently emerged the need to extend classic network analytical methodologies in order to cope with a very important, previously underestimated, semantic information: *time*. Such awareness has been the linchpin for recent works that have started to redefine form scratch well known network problems in order to better understand the evolving nature of human interactions. Indeed, social networks are highly dynamic realities: nodes and edges appear and disappear as time goes by describing the natural lives of social ties: for this reason. it is mandatory to assess the impact that time-aware approaches have on the solution of network problems. Moving from the analysis of the strength of social ties, passing through node ranking and link prediction till reaching community discovery, this thesis aims to discuss data-driven methodologies specifically tailored to approach social network issues in semantic enriched scenarios. To this end, both static and dynamic analytical processes will be introduced and tested on real world data.

The only reason for time is so that everything doesn't happen at once.

— Albert Einstein

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

> Do the difficult things while they are easy and do the great things while they are small. A journey of a thousand miles must begin with a single step.
>
> — *Laozi*

Nowadays Complex Networks are pervasively used to model and describe the behaviors of a wide range of real world phenomena. Social relationships, biological interactions, transportation, commercial exchanges are only few of the several scenarios usually studied with the support of instruments borrowed by graph theory.
Countless problems are formulated, or can be formulated, upon such structures. Moreover, when the semantics attached to nodes and edges as well as the network structural or topological characteristics change the same analytical approach developed to solve a well-defined problem can led to results having slightly different interpretations. Even if formalizing objectively a network problem is almost always feasible, proposing resolutive approaches to it able to guarantee high efficiency and effectiveness regardless the type of network analyzed is a very complex task. In order to avoid such issues researchers focused their effort into the study of peculiar typologies of networks proposing algorithms and analytical tools specifically tailored upon their characteristics and semantic definitions. Among all the fields that emerged in the last decades due to this specialization, Social Network Analysis (henceforth SNA) is the one that makes use of graph mining techniques to understand human behaviors.

Due to the ever growing number of social data available (Online Social Networks, Call graphs, GPS tracks...) the study of human interactions and the formalization of models which explain collective as well as individual activities has increasingly received attention. Moreover, Social mining has rapidly become one of the most interdisciplinary research playground ever: in this scenario, thanks to the advent of the so called *Big Data* as well as to the development of accurate physical models and statistical analysis tools several sociological theories (i.e. *"Strength of the ties"*, *"Dunbar number"*, *"Six degrees of separation"*...) were confirmed and disproved.

Nowadays countless aspects of human sociality are analyzed in order to gain useful insights and to make predictions on individual as well as on group activities, preferences and habits. Furthermore, among the most interesting categories of problems studied so far within the SNA scientific community we must recall the following ones:

- *Individual* analysis: this first set of open problems involves to the study of the *local structures* that compose complex networks. To this category belong all the problems aimed to understand local properties of simple network entities (i.e. nodes and edges). The major research

themes that fall in this class are: (i) Link Prediction, i.e. the forecast of edges that are likely to appear in the future given the actual state of a network; (ii) Node Ranking, i.e. the identification of the most important nodes w.r.t. a given query or set of characteristics; (iii) Tie strength analysis, i.e. the identification and assessment of the strongest bonds among nodes within a network; (iv) Multiplex analysis, i.e. the analysis of networks in which among pair of nodes are present, at the same time, multiple edges carrying different semantics. In the following we will call the problems of this category *structural* as well as *individual*.

- *Collective* analysis: conversely from the previous category, the problems which fall in this set deal not exclusively with the *local* information attached to nodes and edges but analyze well-defined *collections* of entities belonging to a complex networks. Examples of research topics in this area are: (i) Community Discovery, i.e. the decomposition of a network in tightly connected set of nodes; (ii) Homophily analysis, i.e. the identification of social groups homogeneous w.r.t. a given property, or behavior; (iii) Frequent Pattern mining, i.e. the identification and extraction of frequent topological patterns hidden within a network; (iv) Information Diffusion and Leader detection, i.e. the identification of the sets of nodes that are able to generate information cascades. In the following we will call the problems of such category *topological* as well as *collective*.

The overwhelming number of approaches proposed in the last decades to tackle problems belonging to those categories were able to lift significantly our knowledge on the laws that regulate complex phenomena: interaction patterns were discovered, strategies to suggest behaviors as well as pieces information to social service users proposed, innovative ways to forecast flu (as well as content) spreading on a given social tissue studied. However, the main focus of research has concerned the analysis of static realities: only a small ensemble of works have tried to understand the underlying dynamics of real phenomena or, at least, to exploit them in order to provide more accurate analytical results.

A non negligible part of the networks studied within the SNA community are built upon dynamic, rapidly evolving, human relationships: friendships ties, working collaborations, spatial co-locations are all examples of simple bonds among people that, as time goes by, can rise, evolve and fall. Classical approaches try to simplify the analytical process by flattening evolutive behaviors, by observing only a static picture representing a single moment of the network history. The reason behind this choice has to be identified in a *QSSA* (Quasi Steady State Assumption): networks can, somehow, be "frozen" in time because the perturbations in their topology occur only in the long run. Unfortunately this assumption holds only in very particular cases, especially when dealing with social scenarios. A clear example is given by call graphs, in which the interactions among users appear as a continuos flow and whose social strength decrease as time goes by: in such context, we can easily picture the degree of oversimplification an analysis performed under QSSA introduces.
For this reason, several strategies aimed at allowing dynamical modeling of networks have been recently proposed. A first step on the road that starting from the analysis of completely static networks brings to the study of fully dynamic ones is achieved by the introduction of temporal discretization (i.e. the representation of a dynamic network through a set of static, temporal annotated, snapshots). Such reduction represents a first approximation towards a complete dynamic modeling of streamed social interactions: restrict the QSSA constraint independently to each, time-bounded, network snapshot is a way to obtain more realistic and reliable outcomes from analytical processes aimed to describe phenomena related to complete network history. However, defining the right temporal granularity for network partition it is not an easy task due to its context dependency w.r.t. the network semantics (i.e. it is likely that interactions on a call graph posses different degradation time or social strength than the ones that relate users in an Online Social Network). For this reason, recently have started to appear works that analyze dynamic networks as composed by timestamped interactions produced by continuos streams (thus avoiding temporal discretization).

Starting from these observations, in thesis will be presented several resolutive approaches to

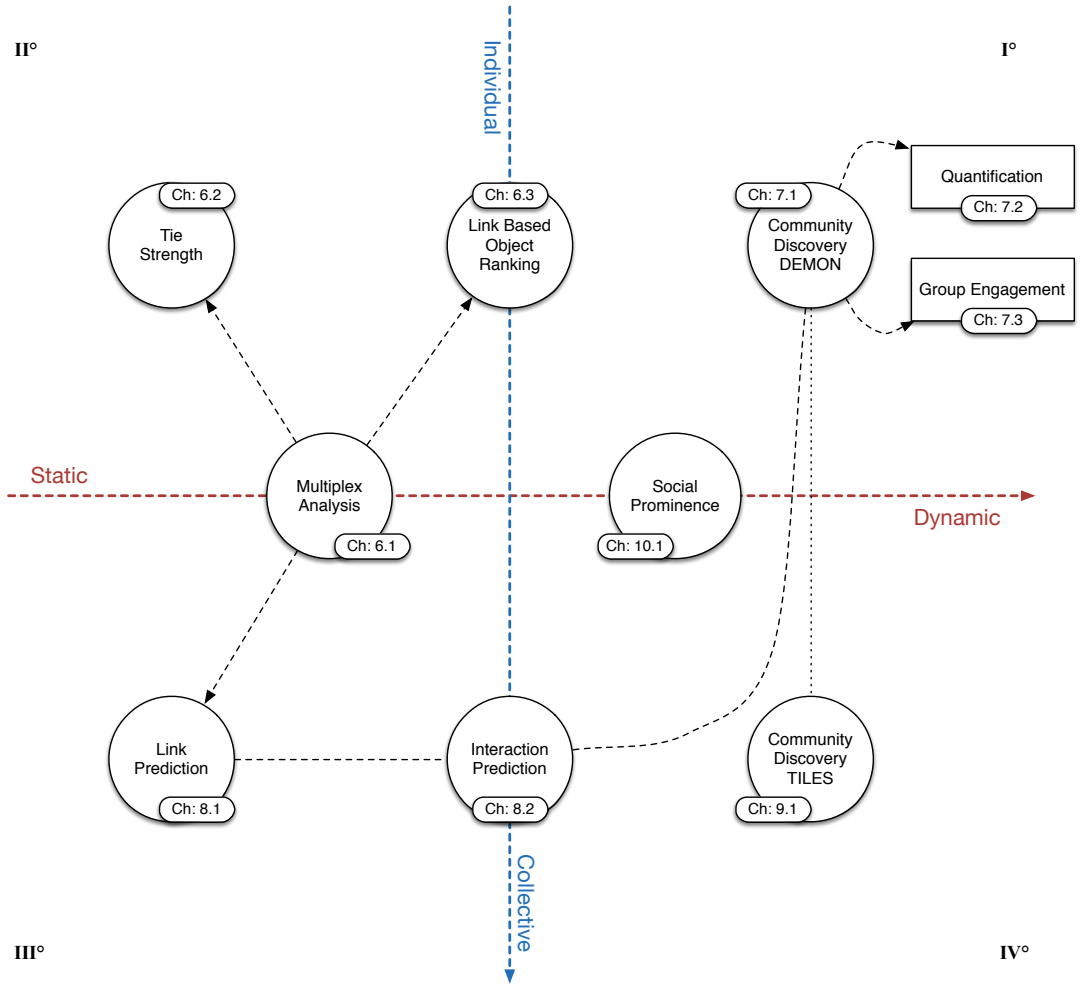


Figure 1.1: From static to dynamic: individual and collective analysis. Analyzed network problem and their classification w.r.t. the static/dynamic and individual/collective dychotomies.

classic network problems. As shown in Figure 1.1 the narrative will develop following two main roads: (i) we will address both static and dynamic scenarios and (ii) approach issues related to network structure and topology. In particular this work is structured in three parts:

Part 1: Setting the Stage.
Here we introduce the preliminary notions needed to become familiar with the field of research covered by this thesis. In Chapter 2 Graph theory definitions as well as classical findings of complex network analysis are introduced and discussed. Chapter 3 introduces the main goal of this thesis: here it is underlined the need of novel analytical models able to tackle classical graph problems from an SNA perspective. Moreover, in this chapter it is discussed how the network dynamics and evolutive patterns can influence such processes and the reasons their study represents a new frontier of Social Network Analysis. Moreover, here we highlight the *structural (individual)* and *topological (collective)* problems which will be addressed in the following parts of the thesis. In Chapter 4 are introduced the related works that cover the applicative scenarios analyzed in the rest of the work: literature regarding Temporal networks, individual and collective analysis and diffusion processes is introduced and the major results highlighted and discussed. Chapter 5 con-

cludes this preliminary part providing insights on some of the real world network datasets that will be referred in the following of the thesis.

PART 2: Social Network Analysis.
In this part are introduced novel approaches able to solve classical complex network problems when no temporal information is taken into account. In Chapter 6 are addressed methodologies developed to solve *structural* network problems. In 6.1 is introduced the context of multidimensional (multiplex) networks, i.e. networks in which multiple semantical annotated connections can, at the same time, exist among two nodes. In 6.2 we introduce, and validate, a new measure able to evaluate tie strength in multiplex networks; in 6.3 we propose UBIK an approach tailored to rank individuals in semantic reach environments expressly studied to solve the human resource finding problem. Conversely, Chapter 7 focuses on the analysis of network *topologies*. Within this chapter, in 7.1, is introduced DEMON an algorithm tailored to approach the Community Discovery problem from a social perspective. The proposed algorithm partitions the network into overlapping substructures searching from denser areas within nodes' ego-networks and then merges them in order to identify tightly connected social groups. Moving from the results obtained by this algorithm in 7.2 is approached the problem of quantifying node labels: here we make use of the homophily observed within communities in order to predict the overall label distribution of uncategorized network nodes. Finally, in 7.3 is analyzed the *Group Engagement* problem: using the massive social network of a VOIP provider, along with the temporal information associated to its users' product usage, we managed to characterize the communities extracted from its underlying social network classifying them by their different level of activeness.

PART 3: Social Networks and their Dynamics.
Moving from the results presented in the previous chapters, this last part aims to reformulate and analyze classical network problems in dynamic scenarios. Chapter 8 covers the analysis of time-aware approaches to network structural problems. In particular, in 8.1 an unsupervised approach to Link Prediction for multiplex networks is discussed while in 8.2 is proposed an extended formulation for the same problem, namely the *Interaction Prediction*. To solve such problem it is proposed a supervised strategy which exploits evolutionary analysis. The introduced analytical process shows how time series forecasting and community information are able to provide meaningful insights on future link formation. In Chapter 9, mirroring the organization of the previous part, will be discussed network *collective* problems. Here in 9.1 an Evolutionary Community Discovery algorithm is introduced. TILES is one of the first approaches able to track communities life-cycles in an online fashion: it follows a *falling dominos* procedure able to, looking only at local network perturbations, update the community structures once a new interaction took place. By definition TILES operates analyzing an edge stream avoiding the need of imposing the temporal granularity for network partitioning. Finally, in 10.1 is introduced a novel approach aimed at characterizing *Social Prominence* on a music related online social network. Analyzing listening data of Last.fm users as well as their social network we were able to define a methodology targeted to identify and characterize the "leaders" which generate information cascades. Moreover, in this chapter we propose a discussion on the structural peculiarities of listening cascades belonging to different music genres. Finally, Chapter 11 concludes the thesis and provide some linchpins for future works.

In Figure 1.1 the two dichotomies (static vs. dynamic, individual vs. collective) are used to organize the works presented in the rest of this thesis into four planes: the upper-left one (II^o quadrant of the cartesian plane) embeds the static-individual problems; the upper-right (I^o quadrant) the static-collective; the bottom-left (III^o quadrant) the dynamic-individual; the bottom-right (IV^o quadrant) the dynamic-collective ones. The contributions presented in the 2^{nd} and 3^{rd} part of this thesis must be seen as steps of a unique path that starting from the static-individual analysis leads to the dynamic-collective one. In detail, moving from the analysis of enriched network structures (multiplex) we discuss the importance that semantic plays in the definition of social contexts. Multiplex networks allow to describe and study how each peculiar type of interaction shapes the connectivity as well as the strength of social ties. Such information will be extensively used in our

works: whereas in 6 the importance of multiplex modeling is discussed, in 7 we define a methodology able to correctly identify communities in a social environment. Our approach, which allows each node to belong at several communities, highlights once more the underlying multiplicity of the social contexts a person can be involved to. Moreover we show that the DEMON communities are able to bound homophily better than other state of art techniques.
Even the temporal informations attached to nodes and edges of evolving social networks can be modeled through the adoption of multiplex graphs. When we allow an observed social phenomena to unfold through time we can exploit such model to reformulate time-aware solutions for complex problems such as the Link Prediction one. The 3^{rd} part of this thesis is specifically dedicated to this complex scenario: the analysis of evolution at *individual* level, studied in 8, allows the observation of micro perturbations which cause widespread mutation at the *collective* scale (analyzed in 9). As time goes by social contexts change, thus we need to approach community discovery using novel time-aware methodologies. Moreover, not only local structure and topology are subject to the action of time: information usually spread on a social graph taking advantage of user-user interactions. In 10.1 we will see how the information content, as well as the prominence of its owner, determine the pattern and strength of its diffusion. Even in this scenario both time and the multiplex semantic will offer a reading key to understand the phenomenon studied.

As shown in Figure 1.1, several of the themes covered in this thesis are tightly linked within each other and, in some cases, need to be classified as "borderline" w.r.t. the individual/collective or static/dynamic dichotomy. In particular we will open the 2^{nd} part of the thesis with the introduction of a model that can be used to represent both static and dynamic semantic enriched network structures. We then cross the individual/collective axis of Figure 1.1 (at the end of Chapter 6) to address the Link-Based object ranking problem which uses topological informations to characterize local structures. Similarly in the 3^{rd} part (dedicated to dynamic analysis), the individual/collective axis is crossed (at the end of Chapter 8) when dealing with the *Interaction Prediction* problem: indeed, our approach make use of network meso-scale topologies (i.e. communities) to predict the appearance of local structures (link/interactions). Finally, we will conclude our analysis by addressing the *Social Prominence* problem in which the network structure will remain frozen in time but is subject to a diffusive process.

The last two parts of this thesis are based on peer reviewed papers published in international conferences. In Part 2, the *individual/structural* analysis, described in 6.1, 6.2 and 6.3, are inherited from [1, 2, 3] while the *collective/topological* ones presented in 7.1 from [4, 5]. Likewise, in Part 3 the *individual/structural* dynamics results presented in (8.1) are gathered from [6] while the *collective/topological* ones discussed in 10.1 from [7]. The papers from which are extracted sections 7.2, 7.3, 8.2 and 9.1 (presented in [8]) are currently under review.

Part I

Setting the Stage

Chapter 2

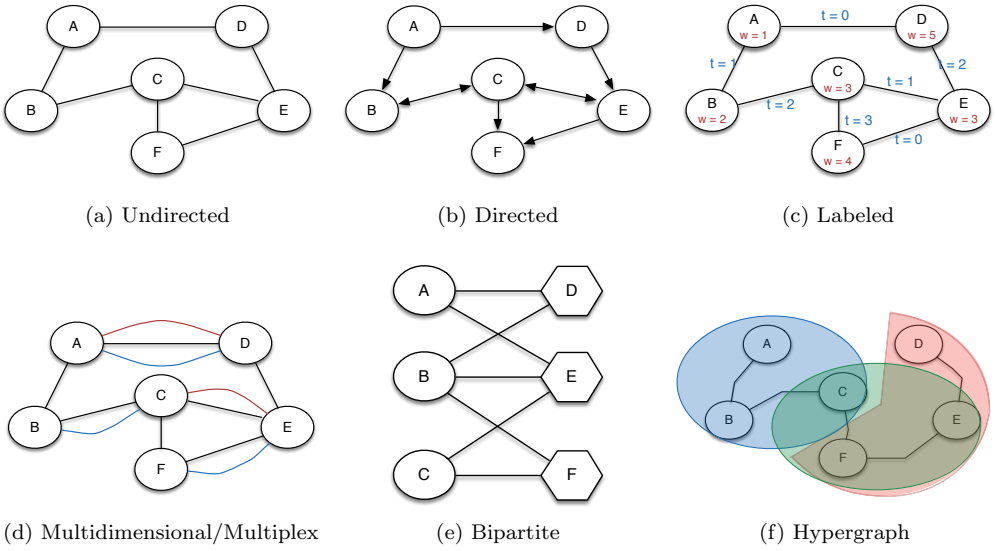
Network Analysis

> The greatest moments are those when you see the result pop up in a graph or in your statistics analysis - that moment you realise you know something no one else does and you get the pleasure of thinking about how to tell them.
>
> — *Emily Oster*

In this chapter we introduce the basic notions of complex network theory. We will start our review by discussing how networks can be represented with graphs (Section 2.1) and which are their peculiar properties (Section 2.2). Subsequently, a short summary of network growth models will be given (Section 2.3): this introduction is intended to provide the reader a preliminary idea of how graph structures evolve and how synthetic dataset can be generated preserving some of the peculiar characteristics observed in real world network.

2.1 The Graph Representation

Graphs are mathematical structures used to model pairwise relations between entities. Usually a graph (henceforth, equivalently referred as a "network") is defined using the notation $G = (V, E)$ where V and E are sets identifying respectively its *nodes*, also called *vertexes*, and the *edges* connecting them, also called *links*. Such structure is highly flexible and allows high expressivity: just enriching its basilar representation we are able to convey more complex knowledge in a easily understandable way. There are several types of both semantic and syntactic extensions of graph structures, here we will recall some of them:



(a) Undirected (b) Directed (c) Labeled

(d) Multidimensional/Multiplex (e) Bipartite (f) Hypergraph

Figure 2.1: Several graph topologies. In the first row are shown, starting from the left, an *undirected*, a *directed* and a *labeled* graph; in the second row a *multiplex*, a *bipartite* graph and an *hypergraph*.

- The edges within a graphs can be undirected or directed: in the latter scenario we will refer to such structures as directed graphs or, more concisely, *digraphs* (or, in absence of cycles *DAG*). As example we can consider a graph representing telephone calls or email messages between individuals: in these cases the graph will be composed of directed edges since each call/message goes only in one direction (i.e. Figure 2.1(b));
- Graphs can be semantically enriched by introducing (possibly multiple) labels on both nodes and edges. As an example we can consider a social graph, i.e. a network in which the nodes identify people and the edges their relationships. As shown in Figure 2.1(c), In such context we might be interested to enrich the graph model with the aim of reporting informations on the type of connection elapsing among pair of nodes or regarding some particular attribute belonging to each vertex (i.e. gender, age. . .);
- Multiple edges (even of different types) can connect, at the same time, the same pairs of nodes: in this case the graph structure evolves in a more complex one giving birth to a *multigraph* (as shown in Figure 2.1(d)). When a multigraph involves edges conveying multiple semantics (i.e., in a social network describing ties of various category such as friendship, co-working, . . .) we will talk about *multiplex* graphs (also known as *multidimentional* or *multirelational* graphs);
- Multiple types of nodes can be related in a graph structure: in this case we are studying *k-partite* graph. An example of *bipartite* graph is the one which relates customers to purchased

goods, as shown in Figure 2.1(e);

- Furthermore, edges can be modeled to bond together more than a pair of nodes at once: in this case we will refer to a more sophisticated structure called *hypergraph* (Figure 2.1(f)).

In the peculiar setting of SNA (i.e., when discussing real world social networks), it is a common practice to make use of the term *entity* or *actor* to refer a network *node*. In the following of the thesis (especially in Part III) we will analyze evolving graphs. Such models are usually obtained by enriching simple, undirected, graphs with edge and/or node labels representing their time of appearance in the network. However, in order to maintain simple the model formulation this enrichment is often avoided by analyzing as its proxy the decomposition of a dynamic network in a graph sequence. We will discuss in depth the most common modeling choices for dynamic networks in 5.2.

2.2 Network Properties

Since the first introduction of the graph model several studies have focused their attention on the formulation and analysis of indicators aimed to characterize its properties. All the issues nowadays studied in the SNA field, as well as in all the other sectors which make use of graph theory as primary investigation tool, are tackled by exploiting such indicators as source of knowledge. In this section we will discuss some properties and indicators used to extract information from real complex networks. In order to provide to the reader the intuition behind the measures discussed in the following we will make use of the toy example in Figure 2.2.

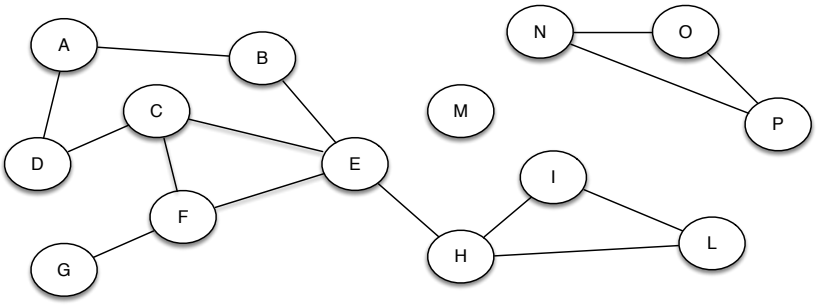


Figure 2.2: Network Properties explainatory toy example.

Degree

The first and most straightforward information that can be analyzed in a networked context is the node **degree**. In graph theory, the degree of a vertex is defined as the number of edges incident to the vertex itself (with self-loops counted twice). Furthermore, in a *digraph* we can distinguish among the *in-degree* (capturing the number of edges entering in a node) and the *out-degree* (which, conversely, describe the number of edges departing from the node). In the rest of the thesis we will mainly analyze undirected network and use $\Gamma(u)$ to identify the neighbor set of a node $u \in V$ and $|\Gamma(u)|$ to denote its degree. While using more complex graph structures (i.e., *multigraphs*, *multiplex* networks or *hipergraphs*) the equivalence of node degree and neighbor set cardinality may fall apart: we will discuss this scenario in 6.1. In Figure 2.2 we can observe, for instance, that node C has degree $|\Gamma(C)| = 3$ and neighbor set $\Gamma(C) = \{D, E, F\}$.

Degree Distribution

Once analyzed the number of edges incident to a given vertex we might be interested to compute the complete **degree distribution** for the nodes in the graph. We define p_k as the fraction of all the vertices in the network having degree of at least k or, equivalently, the probability of choosing uniformly at random a vertex having at most degree k. To represent the degree distribution we can plot the histogram of p_k values for all the values of k in a network G. As we will further discuss in the following section (specifically in 2.3) if we build a graph introducing edges on nodes pair selected uniformly at random we will get a *Poisson* distribution: however, real world networks rarely shows such behavior. In [9] has been underlined that the node degree often follows a more right-skewed distribution showing a long right tail of values that are far above the mean. These networks are called *Scale Free*.
A network may follows a **power law degree distribution** i.e., it can contains a very high amount of nodes with low degree and few hubs with very high degree. This phenomenon can be explained by several theories: the most famous one is the *rich-get-richer* effect which states that, during the network life-cycle the nodes having high degree are more likely to obtain new connections. This idea introduces the concept of node *aging* and the degree of its strength (namely the ratio between the

high degree vertices and the other low degree vertices) can be captured by analyzing the exponent of the cumulative distribution's slope. A power law can be approximated with $p_k \sim k^{-\alpha}$ meaning that the probability that a randomly chosen vertex has degree greater or equal to k follow this function. It has been experimentally proved that in most of real word networks α takes values between 2 and 3 [10].

Connected Components

The component to which a vertex belongs is the set of vertices that can be reached from it by paths running along edges of the graph. All those edges that if removed cause the immediate partitioning of a component in to two disconnected ones are called *bridges*. In a directed graph a vertex has both an in-component and an out-component, which are the sets of vertices from which the vertex can be reached and that can be reached from it. In network theory, a giant component is a connected subgraph which contains a finite fraction of all the graph nodes. Since such concept holds only in the limit of infinite graph size, a giant component would be composed of infinitely many nodes event though the fraction itself could be small.It has been observed that many real world social networks present a giant component, that collects from 70% to 100% of the nodes of the network. Usually, the Giant Component appears when the average degree is greater than 1. In Figure 2.2 we can observe three connected components: $\{A, B, C, D, E, F, G, H, I, L\}$ (the giant one), $\{N, O, P\}$ and $\{M\}$.

Paths

Given two nodes $u, v \in V$, a **path** among them is defined as the sequence of edges that are crossed during a visit starting da u and ending in v. Moreover, with **geodesic path** is identified the shortest path connecting a pair of nodes. Note that there may be, and often there are, more than one geodesic paths between two vertices. In graph theory, the shortest path problem relates to finding a path between two nodes such that the sum of the weights of its constituent edges is minimized. The most simple scenario involves the analysis of a specific instantiation of the problem, in which all edges are unweighted, or their weights are all equal to one. In this case the shortest path is the minimum number of edges to be crossed in order to go from one vertex to another (the *geodesic path*). It has been observed that most vertices pair in real networks seem to be connected by very short paths. This is the so called *small-world* effect [11] (which will be further discussed in 2.3). In practice, the average length of all the geodesic paths in a network is often quite small, much smaller than the number n of vertices belonging to it.
Strictly connected to the path definition is the **diameter** one. The diameter of a network is the length (in number of edges) of the longest geodesic path between any two vertices. In real world networks its value can be approximated with $\log n$ where $n = |V|$. Usually the diameter shrinks as networks grows. This means that if we have a social network and observe new interaction among its users the distance between its most distant entities usually became smaller and smaller [12]. In Figure 2.2 the diameter has length 4 (in particular, an example of longest shortest path is provided by: $A \to B \to E \to H \to L$).

Centrality Measures

When studying a complex phenomenon, identify the relative importance of the agents which participate in it is a frequent issue. In graph theory the concept of **centrality** is used to identify a well defined set of measures aimed at assessing the relative importance of both nodes and edges given their position within the network. Among all the indicators proposed in the last decades three are certainly the most used: **betweenness**, **closeness** and **eigenvector** centrality.

- The **betweenness** centrality of a vertex is the number of geodesic paths between other vertices that run through it. Similarly, this measure can be defined on edges: in this case it relates to the number of geodesic path crossing a given link. Some studies have shown that the node betweenness distribution follows a power law for many networks [13]. Betweenness

centrality can also be viewed as a measure of network resilience: it tells us how many geodesic paths will get longer when a vertex (or edge) is removed from the network.

- The **closeness** centrality is another centrality index which computes the average distance of a vertex from every other vertex in the network. Obviously, in case of a graph composed by more than one component this definition need to be revised in order to consider the distances among the vertices connected by a path.
- Finally, **eigenvector** centrality assigns relative scores to all nodes in the network exploiting the idea that connections to high-scoring nodes contribute more to the score of the node in question than equal connections to low-scoring nodes. Google's PageRank [14] is a variant of the Eigenvector centrality measure. Another closely related centrality measure is Katz centrality.

Transitivity

Another measure able to explain relevant phenomena of real world networks is the **transitivity**, also known as **clustering** coefficient. This indicator captures the so called *"triadic closure"* effect: it computes the ratio of closed triangles over all the triplets of nodes. Its final goal is to unveil the degree to which nodes in a graph tend to cluster together. Two versions of this measure exist: a global and a local one. The former version was designed to give an overall indication of the network clustering, whereas the latter gives an indication of the embeddedness of single nodes. A graph is considered small-world, if its average clustering coefficient $\bar{C}$ is significantly higher than a random graph constructed on the same vertex set, and if the graph has approximately the same mean-shortest path length as its corresponding random graph. Local clustering coefficient is defined as:

$$CC(u) = \frac{\#closed\ triangles}{\#triplets} \tag{2.1}$$

As an example, in Figure 2.2 we have:

$CC(M) = CC(N) = CC(P) = CC(I) = CC(L) = CC(O) = 1,$
$CC(C) = CC(F) = CC(H) = \frac{1}{2},$
$CC(E) = \frac{1}{3}$
$CC(D) = CC(A) = CC(B) = CC(G) = 0$

Community Structure

One important characteristic of Social Networks, deeply connected with the aforementioned transitivity, is the presence of **community** structures, i.e., groups of vertices that have a high density of edges within them and a lower density in between. The presence of such mesoscale structures are often explained by the observation that people tend to cluster in homogeneous, tightly connected, groups. Moving from this, several works (for a detailed survey see [15]) have tackled the problem of identifying, within complex networks, hidden communities that satisfy specific topological characteristics. In social networks it is straightforward to verify that people do organize themselves into (possibly overlapping) groups w.r.t. one or more shared characteristic (i.e., interests, occupation, age...). In this thesis we will further discuss the Community Discovery problem in 4.2.1 and, later on, we will propose two algorithms able to provide to it a solution both in the static (Demon, 7.1) and dynamic (Tiles, 9.1) network scenarios. Observing Figure 2.2 we can notice some dense areas that can identify distinct communities: $\{A, B, C, D, E, F, G\}$, $\{H, I, L\}$, $\{N, O, P\}$ and $\{M\}$.

2.3 Network Models

In this section we present the most common models for synthetic graphs. The general aim of network modeling is to capture some essential properties lying behind real-world phenomena and replicate them while generating synthetic data, all imposing only few simple coinstraints (i.e., the *rich-get-richer* effect explanation for power law degree distributions). As we will see each model focuses only on few properties of real world complex networks and tries to describe how and why they arise and providing a way synthesize structures which respect them.
The proposed overview covers some of the most studied network models: Random Graphs (subsection 2.3.1), Small World Networks (subsection 2.3.2), Scale Free Networks (subsection 2.3.3) and Forrest Fire model (subsection 2.3.4).

2.3.1 Random Graphs

One of the most famous model produced in the 20th century is the Random Graph one. Introduced, independently, by Solomonoff and Rapoport [16] and by Erdös and Rényi [17] the random graph theory is one of the breakthrough that started the analysis of complex networks topologies and characteristics.
A random graph, $G_{n,m}$, is dened as a set of n labeled nodes connected by m edges chosen randomly, with probability p, from the $\frac{|V|(|V|-1)}{2}$ possible edges. Such definition implies the existence of several different graphs for the same chosen value of n and m, all of them belonging to a probability space in which every realization is equiprobable.
Since its first formulation several mathematical studies have analyzed its theoretical implications. Some of the major results on random graphs have concerned the observations of:

- their tendency to have small diameter;
- their tendency to have degree distribution that follow a *Poisson*;
- their tendency to show very low clustering coefficient;
- the similarity among their diameter value and their average path length;
- the presence of a giant component if their mean degree $\langle k \rangle = pn < 1$ and, conversely, the presence after a phase transition for $\langle k \rangle > 1$ of isolated trees.

This model, born in 1959, was one of the first attempt to describe, through the graph theory, social and communication networks. The assumption that real networks have to shown Erdös and Rényi topologies was subverted by numerous measurements done at the end of the last century on real networks: such observations became the trigger for the study of alternative models.

2.3.2 Small World

In a paper published in 1967, the sociologist Stanley Milgram proposes an experiment [18], that became very popular thanks to a play by John Guare [19], and known nowadays by the name of *"Six degrees of separation"*. In his work Milgram decided to verify whether the small-world experience (e.g., an unknown person we meet knows a person we know) is related to some real phenomenon or is only a simple anecdotes. For his experiment the sociologist asked Midwestern volunteers to send packages to a stranger in Boston knowing only his name and profession: packages could not have been sent directly to the recipient, but should have been delivered only by exploiting the personal contacts. The results shown that, on average, the number of intermediaries needed to complete the chain was 5.51. This finding led, in 1998, Watts and Strogatz [11] to describe a peculiar kind of networks as *small-world networks*, in analogy with the small-world phenomenon. In their work, they analyzed the neural network of the worm *C. Elegans*, the collaboration graphs of film actors as well as the western US power grid and found that all these networks exhibit a small average path length and a high clustering coefficient. The latter observation was in contrast

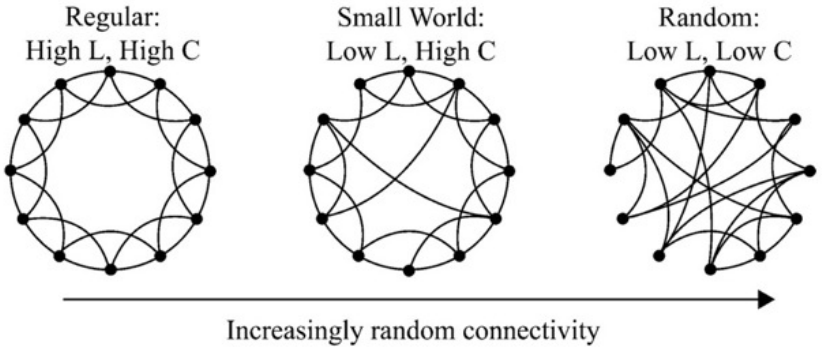


Figure 2.3: Small World connectivity features.

with the characteristic low clustering coefficient shown by random graphs since it recall the more rigid structure typical of regular lattices.
Due to this observation, Watts and Strogatz decided to propose a model able to interpolate between the regular structure of lattices and the random network one. The iterative approach they used to perform such interpolation was quite simple: starting from a circular lattices in which each node is connected to k neighbors, they rewire with probability p an edge at a time avoiding duplicate edges and self-loops. In a variant of this model [20], that does not contemplate rewiring, few random edges were added to a lattices in order to lower the average path length (recreating in this way the small-world phenomenon).

Varying the p value in the Watts and Strogatz model we can observe perturbations in the network structure (as shown in Figure 2.3). In particular:

- with $p = 0$ we obtain a regular lattice with high clustering coefficient;
- with $p = 1$ we get a random networks that exhibit small-world properties and have low clustering coefficient;
- for values of p that lies within the interval $(0, 1)$ we observe networks having high clustering coefficient together with small geodesic distances among their nodes.

Given the value of p, Watt and Strogatz analyzed the clustering coefficient $C(p)$ and the characteristic path length $L(p)$. They observed that: (i) increasing the value of p also the number of cases for which $L(p)$ is almost small as L_{random} increase and (ii) at the same time $C(p) >> C_{random}$. This effect was justified with the observation that the rewired edges were able to introduce random shortcuts connecting nodes otherwise distant in the original lattice.

The highly clustered nature of networks was guessed in social context by Granovetter. He proposed an excellent sociological interpretation: (i) the social network that shapes our society is composed of small tightly connected circles of friends and (ii) such circles are connected among them by weak social ties. In this scenario, the rewired edges introduced by Watts and Strogatz act as the weak ties which connect members of the societal cliques to their not-so-close acquaintances.
The small-world effect and the clustered nature of real networks were discovered in a wide range of natural and artificial structures like the Internet [21, 22], the WWW [23, 24], the email contacts networks [25], cellular networks [26], scientific collaboration networks [27, 28], in neural networks [11, 29] as well as in the Messenger contacts network [30]. These findings suggest that the observations made by Milgram and Granovetter are not bound to social networks but, instead, have broad application in both natural and technological contexts.

2.3.3 Scale Free

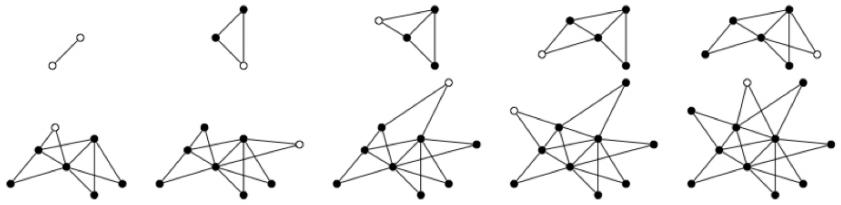


Figure 2.4: Scale Free network growth as proposed by Barábasi-Albert model. Starting from a dyad (top-left) at each itaration a new node, hilighted in white, and two new edges were added following preferential attachment. Older nodes becames hubs as time goes by.

An aspect not taken into account by the previous models regards the presence of *hubs* among the nodes: a hub is a node that is has an higher degree w.r.t. the majority of the nodes belonging to the same network. Hubs were observed in almost all real world networks: a well-connected user in Facebook, a celebrity in Twitter, a major airport (as the international ones in Rome, New York or Frankfurt) a well-known scientist in a collaboration network are all examples of such particular kind of nodes.
In order to study this property it is used the distribution $P(k)$ (already introduced in 2.2). As already discussed, the Erdös-Rényi model imposes a *Poisson* distribution for $P(k)$: however, in many real networks $P(k)$ appears to be highly skewed and to decay much more slowly than a Poisson, mostly following a *Power Law* $P(k) \simeq k^{\gamma}$. Networks which follow such degree distribution are called scale-free, because they do not have a scale, that is a characteristic node.
Although this distribution is not universal in networks [31], it is very common and it can be observed in several artificial and natural networks, such as the WWW [24], Internet backbone [32] and the metabolic reaction networks [26]. On the other hand, in scientific co-authorship networks the degree distribution is well fitted by a *Power Law* with an exponential cutoff, for the US power grid it is an exponential distribution while for a social network of Mormons in Utah, $P(k)$ is a *gaussian* [31].

In 1999, Barabási and Albert [33] showed that this heavy-tailed degree distribution is due to networks continuos expansion by the addiction of new vertices. The two scientists speculate that new nodes preferentially attach to ones that are already well connected in such a way that "richer nodes became richer". On top of this two assumption, the growth over time of networks and the preferential attachment, they proposed the *Scale Free* model (also known as Barabási-Albert model). Starting at a time t_0 with a low number of nodes, at every time step a new node is added with m edges that link to different vertices already present in the network: the probability of choosing a specific target node to connect with is proportional to its degree (an example is shown in Figure 2.4). After t time steps the model converge to a random network with $t + m_0$ nodes and m_t edges. This approach led to a scale-invariant state characterized by a heavy tail degree distribution having exponent $\gamma = 2.9 \pm 0.1$. Due to the huge impact caused by this model, more sophisticated variants were proposed during the last decade in order to include, for instance, the effects obtained by adding link rewiring, node aging (which imposes that when reached a certain degree of maturity a node cannot accept new links), or varying the rules describing the preferential attachment process.

2.3.4 Forest Fire

Once characterized the Scale Free and Small World phenomena, several models were proposed in order to describe, and build, networks having both those desirable properties. One of them, probably the most famous, is the *Forest Fire*[12] one. Proposed by Leskovec, this growth model tries

to introduce community structure (which appear in almost all the social-like real networks) into the generative process. In particular, the authors' aim is to simulate the shrinking diameters that have been observed in complex networks, i.e., the fact that real networks tend to have heavy-tailed distributions for in- and out-degrees and a densification power laws which makes them become denser over time. The latter condition strictly relates to the community structure, previously taken into account by other models (i.e., copying model [34, 35]).
In order to perform this task, Leskovetc defines a basic version of the model in the following way: nodes arrive one at a time and form out-links to a subset of the earlier nodes; to form out-links, a new node v attaches to a node w in the existing graph and then it begins burning links outward from w, linking with a certain probability to any new node it discovers. This process could be intuitively seen as the strategy by which an author of a paper identifies references to include in the bibliography. He or she finds a first paper to cite, chooses a subset of the references in that paper (modeled here as random), and continues recursively with the papers discovered in this way. Depending on the bibliographic aids being used in this process, it may also be possible to chase back-links to papers that cite the paper under consideration. Similar scenarios can be considered for social networks: a new computer science graduate student arrives at a university, meets some older students who introduce him/her to their friends and the introductions may continue recursively. Adding edges in this way, each node connecting only with entities that are closer to its center of gravity, the community structure arise very quickly.

Chapter 3

Complex Networks and Time

> We must use time wisely and forever realize that the time is always ripe to do right.
>
> — *Nelson Mandela*

One of the most challenging task in data mining regards the analysis of temporal annotated data. Temporal annotated sequences can be built from data produced by a wide spectrum of human related processes ranging from social interactions to mobility traces and financial operations. The ability to extract information from such timestamped observations became crucial when we need to make inference on the behaviors of time-evolving complex systems. Without loss of generality, we can state that the ultimate goal of *temporal data mining* is to discover hidden relations between sequences and subsequences of events in order to describe and/or forecast the behaviors of the observed phenomena.

Network mining problems do not differ from other classical data mining ones. A wide range of analysis can be performed on static networks, however the phenomena we are used to observe, as well as the world we live in, are constantly evolving: as time goes by relationships change, links are substituted by new ones, new collaborations arise and old ones fall apart.
Freezing networks in time is certainly very useful in order to observe, study and categorize some of their peculiar traits but it is not enough if we are interested in understanding the dynamics that regulate their lives. Hence, for some of the most interesting network problems can be provided augmented formulations that take into account the role played by time.
Looking at the plethora of evolutionary issues nowadays studied in the complex networks field, we can build a simple, even if not extensive, classification:

- *Individual Dynamics:*
  In this category fall all those problems which analyze how local structures, edges and nodes, rise and fall (i.e. Link/Node Prediction, Dynamic Vertex Coloring...);
- *Collective Dynamics:*
  Here the main focus is analyzing how topology changes when local structures do (i.e. Evolutionary Community Discovery, Frequent Patterns...);
- *Diffusion Processes:*
  To this last category belong all those tasks whose aim is to understand how information flow on network structures (both in the case of static and dynamic structure/topology);

This three families of problems can be logically seen as a hierarchy as shown in Figure 3.1: *Individual Dynamics* analysis tries to explain how, at a local level, nodes connect and how existing

links vanish or strengthen through time; *Collective Dynamics* analysis models the behaviors of set of nodes and edges; *Diffusion* looks at the whole network in order to understand how its structure and topology facilitate (or make it difficult) the spreading of "informations".

Moreover, as already discussed in chapter 1, we can think to *individual/structural* and *collective/topological* analysis as part of a dichotomy. Since in this particular dichotomy *"tertium non datur"* do not holds (several applications can benefit by mixed structural-topological approaches), we can picture diffusive processes as examples of tasks in which individual and collective behaviors are exploited in conjunction in order to model more complex phenomena.

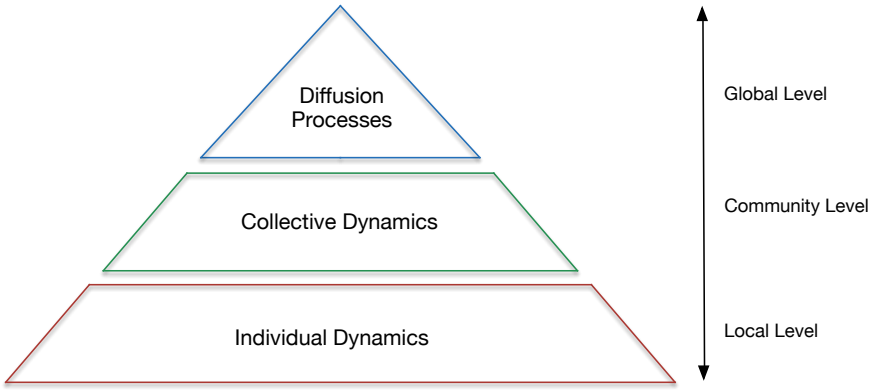


Figure 3.1: From the local to the global level: dynamic network problems and their classification.

The dependences from these three *layers* can be tackled in several ways: (i) they can be *"ignored"* in order to propose models that fit a specific purpose and provide a straightforward, easily understandable, answer to a specific problem (i.e., simulate virus spreading on a static network); (ii) they can be *"assumed"* in order to simulate dynamic behavior using as starting point data without temporal annotation (i.e., imposing an underlying network growth model to produce new node/edges when forecasting new interactions); (iii) finally, they can be directly *"exploited"* by embedding in the analysis the real dynamics expressed by the data (i.e., tracking community evolution using as input the real sequences of node and edges appearance and disappearance). All these approaches carry advantages as well as drawbacks. Working with fine grained data without making abstractions on general behaviors ("let the data talk") will produce very realistic results that can't be transferred from the analyzed domain to a different one. Conversely, adopting a more general framework will lead to less precise approaches which can be easily adapted to a wide range of phenomena. In the following chapters, whenever possible, we will provide analytical tools guided by the knowledge extracted from real data more than frameworks based on general models.

Moreover, in order to be tackled with a data mining perspective, the analytical tools we design demand the availability of temporal annotated network datasets. Fortunately, thanks to the ever growing availability of *Big Data*, nowadays semantic rich social tissues can be in depth analyzed: online social networks, as well as telephone call and mail exchange logs, are valuable resources that can be used as proxies to estimate offline social dynamics. When dealing with social data understand the role time plays to shape structure and topology of a network lead to a deeper comprehension of the dynamics expressed by the observed context. Individuals act independently one from another and, at least to some extent, can be described by their own local information (i.e., their characteristic interaction time as well as the structure of their ego-network): however, like neurons in a nervous system, their actions often concur to the specification of precise and more complex group-related functions (i.e., they can establish collaboration or organize themselves in self regulated organizations/social circles). All these dynamics show a common leitmotif: they are highly complex and difficult to forecast and model.

In this thesis we will analyze a subset of problems for each of the identified classes and present novel approaches designed to solve them. Indeed, we will describe analytical processes aimed at solving both static and dynamic formulations of a well-defined set of network problems: from chapter to chapter we will highlight the impact that the temporal dimension has on the results we were able to obtain.
In order to do that we will narrow our field of investigation from the general complex networks analysis to the more specific field of the SNA. Thanks to the ever growing usage of online (and offline) communication services, social ties are among the most volatile structures that we are able to track and observe: for this reason they represent the perfect fit for a combined static and dynamic analysis able to unveil some of the leading traits of evolving interconnected structures. The choice of this particular setup, as will be underlined in Chapter 4, is often pursued in literature since (i) it makes possible to provide ad-hoc models that solve network problems using real world data and (ii) it allows the validation of the obtained results through the adoption of well-established sociological theories which are often able to explain the observed phenomena.

Once defined our settings, in the following sections we present a preliminary introduction to the *structural* and *topological* network problems (respectively in 3.1, 3.2) that will be extensively discussed in the rest of this thesis.

3.1 Local Structures

Among all the network related topics studied in the last decades, the analysis of nodes and edges local surroundings is the one that has attracted ever growing attentions. Henceforth, we will call the problems belonging to this category "*structural*" (or equivalently *"individual"*) since their aim is to understand how local structures can be exploited in order to unveil meaningful knowledge at the node/edge level. In this section we will briefly introduce three problems that will be addressed in the rest of the thesis: *Ties strength*, *Link Prediction* and *Link-Based Object Ranking*.

Ties Strength

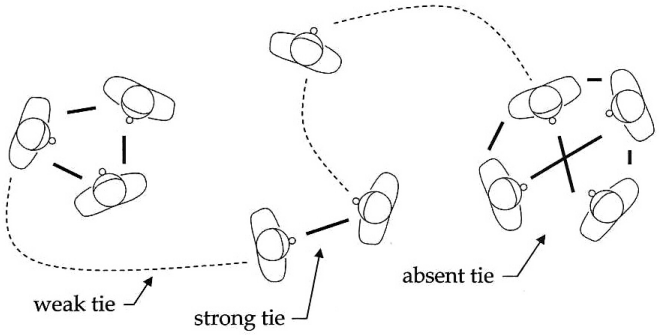


Figure 3.2: Graphical representation of tie strength in a social context.

The advent of OSNs (Online Social Networks) has completely redefined the way we conceive social relationships: indeed, such instruments have broken the constraints of time and geography that limited our social world. In these virtual environments establishing new friendships is immediate and effortless, so it is reasonable to think that the number of our social bonds could approach to infinite, removing the social boundaries of our modern, technological era. However, OSNs have allowed us to build massive networks of weak ties: acquaintances and non intimate ties we use all the time to reach out persons, business requests, speaking engagements, or ideas and advice. Despite such enormous quantity of acquaintances, recent works have revealed two major aspects online real social networks that mirror real world social contexts:

- people still have the same circle of intimacy as ever [36, 37],
- the formation of friendships is strongly influenced by the geographic distance, thus breaking down the illusion of living in a "global village" [38, 39].

People users tend to interact intensely with a small subset of individuals, carrying out social grooming in order to maintain and nurture strong, intense ties. Strong ties connect us with the friends we really trust, people whose social circles tightly overlap with our own and, often, they are also people that are most like us. Although such trusted friendships are not so important in the spreading of information [40], new ideas [41], or for finding a job [42], they can affect our emotional and economic support [43, 44] and often join together to lead organizations through times of crisis [45]. Unfortunately, the majority of social media do not incorporate explicitly tie strength notions during the creation and management of relationships, and treat all their users the same: friend or stranger, with little or nothing in between. A first attempt to take into account the social role of friendships was done by Facebook and Google+ by the introduction of the so called "circles". Users of such platforms can use circles as a way to organize their contacts in a sort of address book, creating different groups for relatives, work colleagues, close friends and so on. However, such conceptual organization of contacts does not provide quantitive information about the real strength of the ties, but only the nature of relationships between users and the context in

which they take place. For example, the presence of a user in the circle of work colleagues does not necessarily imply the existence of a strong tie, and does not provide any explicit quantification of the importance of his relationships with other members of the same group. Actually, it does not exist a formal, unique and shared definition of tie strength, and literature has often provided very personal interpretations of Granovetter's intuition: *"the strength of a tie is a (probably linear) combination of the amount of time, the emotional intensity, the intimacy (mutual confiding) and the reciprocal service which characterize the tie"* [46]. The most frequently used measurements of tie strength in social networks are based on the number of conversations between users [36], or, in the mobile phone context, on the duration of calls [40]. However, in our opinion these common approaches suffer two major shortcomings. Firstly, the number and intensity of conversations strongly depend from user to user, making it difficult to understand which of these conversations are dedicated to intimate relationships. Secondly, they do not take into account that strong ties must be powered by a form of social grooming, that is mainly based on geographical proximity and face-to-face contacts.

Link Prediction

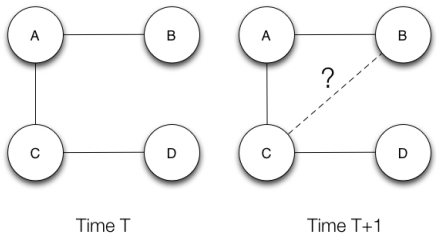


Figure 3.3: Link Prediction: an example.

Reasoning on networks evolution a very urgent question to answer arises: *is there any rule that regulates the rising of new edges?* or similarly, *there exists couples of nodes that are most likely to establish a connection than the others?*
As we have seen before, a wide set of models were studied with the aim to understand and reproduce real networks traits: all those models are generative, and mimic the processes of networks growth over time. Here we are interested to address a slightly different issue: we know the nodes of our network (we assume that no other nodes could be added in successive time steps) and want to study the probability that two of them became neighbors in the future. Suggest new friendships on a social network, co-authorship on a professional network or interesting products of an online-market are certainly facilities that online services nowadays need to offer to their users. Link Prediction group tighter all those problems: it is defined as the problem of identifying, given a snapshot of a network G at a time t_0, the top-k edges that are most likely to appear among its set of nodes, at a time t_1, restricting the prediction to those nodes that are not connected by edges during the first observation.
Correctly predict a new link in a (often) sparse network is a hard task to accomplish: for this reason several approaches were proposed in order to study this evolutive aspect of complex networks, using both supervised and unsupervised methodologies. In particular, unsupervised approaches are built upon on local (neighborhood-based), or global (path-based), topological aspects relative to the pairs of nodes for which a prediction is needed. Those methodologies, given their simple nature, were shown to be able to guarantee around 10% of correct predictions in several OSN datasets: given the complexity of the problem, this value that could seem very low is actually a very good result. In order to improve such results, supervised approaches that exploit not only network topology but even semantic informations attached to its nodes (and edges) were proposed: we will discuss them in detail in 4.1.2.

Link-Based Object Ranking

Among the most prominent issues in network analysis, surely the most famous one among the link mining tasks, is the link-based object ranking (LBR). The objective of LBR is to exploit the link structure of a graph to provide an ordering (i.e., a ranking) of its vertexes. In the context of web information retrieval, the PageRank[14] and HITS[47] algorithms are the most notable approaches to LBR.

- PageRank simulates a web surfing with a random walk where the surfer randomly selects and follows links and, with a certain probability, jumps to a new web page to start another traversal of the graph. The rank of a given web page in this context is given by the fraction of time that the surfer would spend in it if the random process were iterated ad infinitum. This can be determined by computing the steady-state distribution of the random process.

- HITS assumes a slightly more complex process, modeling the web as being composed of two types of nodes: hubs and authorities. Hubs are web pages that link to many authoritative pages. Authorities are web pages that are linked to by many hubs. Each page in the web is assigned hub and authority scores (which get the same value in the limit case of an undirected graph). These scores are computed iteratively by updating the scores of a page looking only to the scores of the pages in its immediate neighborhood.

Within the SNA, LBR is considered a core analysis task. In this context, the final goal become to rank individuals in a given social network in terms of a measure of their importance, referred to as centrality. As discussed before, measures of *centrality* have been the subject of research in the SNA community for decades. These measures characterize some aspect of the local or global network structure as seen from a given individuals position in the network. Several social scenarios can benefit from ad-hoc ranking approaches: expertise finding, collaborative filtering and recommender systems are only few examples.

Multidimensionality

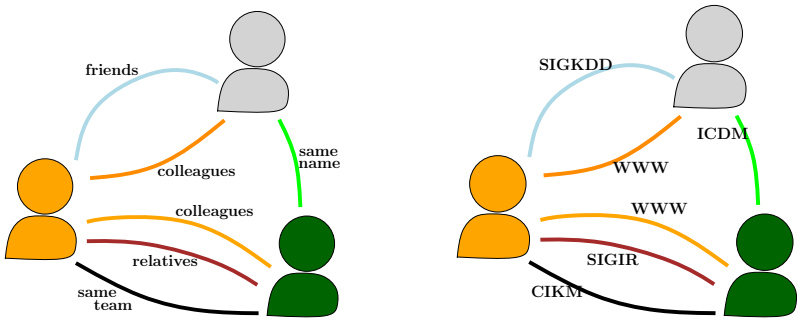


Figure 3.4: Example of multidimensional networks

In recent years, complex networks have been receiving increasing attention by the scientific community, also due to the availability of massive network data from diverse domains, and to the outbreak of novel analytical paradigms which pose relations and links among entities, or people, at the center of investigation. Inspired by real-world scenarios (i.e social networks, technology networks, the Web, biological networks) wide, multidisciplinary, and extensive research has been devoted to the extraction of non trivial knowledge from such complex topologies. Most of the networks studied so far are monodimensional, i.e., networks in which is allowed a single edge between each pair of nodes. In this context, graph analytics has focused to the characterization and measurement of local and global properties (such as diameter, degree distribution, node/edge centrality, connectivity) and to the formulation of more sophisticated problems (i.e., all the issues

based on graph mining and aimed at finding subgraph patterns).
However, in the real world, networks are often multidimesional, i.e., there might be multiple connections between any pair of nodes (two examples in Figure 3.4). Therefore, multidimensional analysis is needed to distinguish among different kinds of interactions, or equivalently to look at interactions from different perspectives.
Dimensions in network data can be either explicit or implicit. In the first case the dimensions directly reflect the various interactions in reality; in the second case, the dimensions are defined by the analyst to reflect different qualities of the interactions that can be inferred from the available data. Those complex systems are referred also as multislice, networks with explicit dimensions are named multiplex, and, often, temporal information is used to derive dimensions for the network where other semantics are not available.
Examples of networks with explicit dimensions are social networks where interactions represent information diffusion: email exchange, instant messaging services and so on. An example of network with implicit dimensions is an on-line social network with several features: in Facebook, while the social dimension is explicit, two users may be connected implicitly by their like on posts of shared friends or by their favorited pages or groups they belong to. Moreover, different dimensions may reflect different types of relationship, or different values of the same relationship.

3.2 Topologies

Moving from the analysis of local network entities, a second and equally relevant field of research regards the study of more complex network *topologies*. Henceforth, we will call the problems belonging to this category "*topological*" (or equivalently *"collective"*) since their aim is to understand how to unveil coherent patterns hidden within complex networks and how to use them in order to make inferences on the behaviors of well defined sets of nodes and edges. In this section we will briefly introduce three *graph mining* problems that will be addressed in the rest of the thesis: *Community Discovery*, *Social Engagement* and *Network Quantification*.

Community Discovery

The problem of extracting communities from a graph, or of dividing its vertexes into communities, has been approached from several perspectives [15]. Algorithms for community extraction have appeared in practically all scientific fields: starting from social network analysis to physics, and computer science. This multiplicity of approaches is probably due to the absence of a formal and shared definition of network "community". Indeed, each algorithm proposed so far extract coherent network substructures which maximize a given topological objective function: however, in the last decades, several objective functions where proposed in order to evaluate the goodness of network partitions leading to the rising of several families of CD approaches. Among the most frequently used evaluation function we can recall: *modularity*, *clustering coefficient* and *conductance*.

From a SNA perspective, communities are perhaps the most basilar bricks that make possible the analysis of complex phenomena: indeed, being able to identify tightly connected sets of nodes allow an in depth analysis of the human sociality. Thanks to the increasing availability of online social network data, we have witnessed the appearance of a wide number of algorithms tailored to capture specific characteristics expressed by human interactions. Such profusion of methodologies has given the opportunity to validate and disprove several sociological theories. For instance, observing the social structure of several OSNs was noticed that the neighborhood of a single node is often composed by a huge number of peers belonging to different semantic contexts (i.e., school, work, sport related...). However, not all the node's acquaintances can be considered as "real" ones: conversely from the real world experience, in online services the action of establish a new ties has no cost: thus, most of the links observed in an OSN does not represent relations existing in real life. Hence, CD solutions in ONS contexts need to take care of both this observations: social groups need to be homogeneous w.r.t. some semantic information (when available) and to avoid the overestimation of user's sociality.

Moreover, as time goes by people tend to modify their social relationships: travels, job changes, rising of new interests are only few examples of the causes that lead to a perturbation (and in some case even to the ending) of the interactions and connections. This dynamic evolution is mostly evident on the social tissue described by communities. Obviously the data produced by OSNs are not expected to describe the evolution of a real world community. Online the timing for the born and ending of a relationships are quicker than the ones we are used to in our daily experience: nonetheless we can speculate that the mechanisms that regulate evolutions are, to some extent, alike. Nowadays, community evolution tracking and forecasting represents one of the most challenging task in the dynamic network analysis field.

Social Engagement

The increasing availability of Big Data describing customers behavior has changed the way Companies advertise their services. Online shopping site as well as OSNs are nowadays collecting data on their users activities and sociality in order to extract information which can guarantee them an edge on competitors.
In this scenario, being able to identify how much users are engaged to a specific product/service

offered by a brand (i.e., Skype video call, Facebook chat rooms, Dropbox file sharing, Google online document editing...) is a powerful tool that can be used to drive decision on future commercial strategies. Service Engagement was studied from different point of views: economics, statistics, sociology are only some of the fields that, often with different purposes, have tried to capture the underlying reasons that drive users to adopt a specific product.
Moreover, the success of a product/service is often due to the virality it is able to achieve. In order to broaden the diffusion of a specific product it becomes mandatory identify a fertile ground, a set of potential users that are likely to be interested to it.
Several SNA studies have shown that homophily is a property that can be observed in almost all human social networks: peoples tend to cluster homogeneously by age, location, interests and, more important in our scenario, tastes. Social communities are perhaps the basic bricks that can bound such phenomenon and that can provide an indicator able to shows who to target when programming an advertising campaign.
The study of "Product Engagement" nowadays is evolved in the analysis of "Social Engagement": by leveraging on social ties and homophily novel analytical tools can be proposed both to describe and forecast how a service is used by sets of deeply connected customers.

Network Quantification

Moving from what have been said on Community Discovery and Social Engagement, another interesting problem which poses its grounds on the existence of homophilic behaviors within well defined network substructures is the *quantification* one.

Many real-world applications require to estimate and monitoring of the distribution of a population across different classes. An example of such applications is the important task to determining the percentage (or "prevalence") of unemployed people across different geographical regions, genders, age ranges or even temporal observations. In the literature, this task has been called *Quantification* [48, 49, 50, 51, 52].

Quantification is closely related to *classification*: however, the goal of classification is different, since in classification we are interested in correctly guessing the true class label of each single individual. Instead, in quantification we are interested in classifying individuals with the goal of estimating the class prevalence, so it is not strictly necessary to classify correctly each single individual. Classification and quantification are different because, while a perfect classifier is also a perfect quantifier, not necessarily a good classifier is also a good quantifier. Indeed, a classifier that on the test set generates a similar number of misclassified items in the different classes is a good quantifier because the compensation of the misclassifications leads towards a perfect estimation of the class distribution.

Most of the works that have been proposed so far address the quantification problem taking into consideration data presented in conventional attribute format. Since the ever-growing availability of web and social media we have a flourish of networking data representing a new important source of information. In this scenario an interesting question arises: *how can the quantification be performed on data describing relationships among the entities of a system?*

The impact of quantification techniques for networking data is potentially high: this because today we are witnessing an ever more effective dissemination of social networks and social media where people express their interests and disseminate information on their opinions, about their habits, and their wishes. The possibility to analyze and quantify the percentage of individuals with specific characteristics or a particular behavior could help the analysis of many social aspects. For example, analyzing Facebook or Google+, platforms in which people can set their education level, we could estimate the level of education of a population. Similarly, we could determine the distribution of the political orientation or the geographical origin of the social network population.

3.3 Diffusion of Information

Even when we consider network structures as static objects, not allowing creation and removal of edges and nodes, there are interesting problems that are strictly related with temporal analysis. One of the most famous is the *diffusion of Information* problem. Users in a social network communicate (i.e., in Facebook they can publish on their wall, express their approval through likes and tags on photos and posts of their friends), exchange informations (i.e., in a professional network news on positions openings) search and promote topics and trends (i.e., hashtags in Twitter). Some of these actions performed by users on a social network could generate cascading phenomena: the aim of studies which approach diffusive phenomena is to understand how they are related to social influence, how information spread recursively from a user to his friends and how this process can define tribe or communities. Two main elements that regulate those kind of processes have been studied:

> **Time**: how rapidly actions became viral? Is there a temporal threshold (or number of hops) for which the diffusion process ends? What is the distribution of temporal intervals among "hops" expected for an information to stop its spread?
>
> **Causes**: does every action necessary lead to a cascading effect or some kind of topological features are needed?

Moreover we can identify several real world tasks whose final goal concern the analysis of how informations spread through a network. An examples, which will be further discussed in the 3^{rd} part of this thesis, regards the identification of the most prominent users within a social network w.r.t. their sharing actions (i.e., the users that facilitate the diffusion of a specific music genre in an online music-driven social network).

Chapter 4

Related Works

> The answers you get from literature
> depend on the questions you pose.
> — *Margaret Atwood*

Moving from the descriptions given in 3.1 and 3.2, in this chapter we present the relevant state of the art for the complex network problems analyzed in the rest of the thesis. Moreover, we will focus on the literature regarding network issues which can be formulated both for static and dynamic scenarios.
In 4.1 is discussed the state of art for each *individual/structural* problem introduced in 3.1. Likewise, in 4.2 is reported the state of art for the *collective/topological* ones introduced in 3.2. Moreover, in section 4.3 is proposed a self-contained collection of related works regarding *Information Diffusion*. Finally, in 4.4, a brief review on temporal networks works and applications is provided.

4.1 Local Structures: Individual Entities analysis

The analysis of network nodes and of their immediate surroundings is one of principal key to read and understand a complex system. In 3.1 we introduced three of the most relevant tasks belonging to this category: *Tie Strength*, *Link Prediction* and *Link-Based Object Ranking*. Here we will revise the relevant state of the art for these well-known problems (respectively in 4.1.1, 4.1.2 and 4.1.3). Moreover, in 4.1.4 we will discuss relevant works in the field of multiplex networks and how this enriched network model has been used in order to tackle some classical tasks from a different angle.

4.1.1 Tie Strength

The problem of measuring the strength of ties in social contexts represent a subtle issue which intersects, often unnoticed, a wide range of network problems. From Link Prediction to network Quantification passing through Community Discovery, a vast number of network tasks estimate implicitly or explicitly which edges are the most important for the graph: for this reason ties analysis can be seen as a mandatory step to take before moving ahead on the road that leads to more complex problems.
The concept of *Tie Strength* was introduced by Mark Granovetter in his seminal paper *"The Strength of Weak Ties"* [46]. There he proposed four main factors shaping the strength of a tie: amount of time, intimacy, intensity and reciprocal services. Subsequent research expanded the list adding demographic and socio-economic status [53], emotional support [54] and network topology [55]. In [56], authors used survey data from three metropolitan areas to discover the predictors of tie strength. Onnela et al. [40] utilized the duration of calls as a measure for tie strength, and observed that social networks are robust to the removal of the strong ties but fall apart after a phase transition if the weak ties are removed. Gilbert and Karahlios [57] presented a predictive model that maps social media data to tie strength, reaching the 85% accuracy in distinguishing between strong and weak ties. Recent works on multiplex networks have addressed the problem of identify strong connections among nodes linked via several dimensions: an example is [58] in which the authors analyzed, separately, the degree distributions of the various dimensions, highlighting the need of novel analytical tools for the multidimensional study of hubs.

4.1.2 Link Prediction

Many papers have addressed the Link Prediction problem (hereafter also referred to as LP), proposing both supervised and unsupervised strategies. In order to provide an overview of the solutions proposed so far, we can categorized the most relevant LP approaches into four groups [59]:

- similarity based strategies;
- maximum likelihood algorithms;
- probabilistic models;
- supervised learning algorithms.

In the first group fall all those approaches that define pairwise measures of similarity as predictive scores for node couples. All the links not yet present in the network are ranked according to their scores: the higher the similarity among a node pair the higher the likelihood that a link among them will arises in the future. Despite its simplicity, define a node similarity measure is a non-trivial challenge. A similarity index can be very simple or very complicated, it may work well for some networks while fails for some others. In this group, [60] presented a solution based on the preferential attachment principle, while [61] and [62] introduce models based on the quantitative characteristics of common neighbors. A very thorough survey on similarity based LP approaches is [63], in which the authors empirically compare many different models proposing a way to standard validate the accuracy of the obtained results (Precision-Recall and ROC curves).

The second group of methods is based on maximum likelihood estimation [64, 65, 66, 67] and move their steps following the results empirical studies which suggest that many real world networks exhibit hierarchical organization. These algorithms require the existence of an organizing principle which shapes the network structure in order to identify detailed rules and specific parameters that maximize the likelihood of the observed structure. From the viewpoint of practical applications, an obvious drawback of maximum likelihood methods is that they are very time consuming. In addition, maximum likelihood methods are probably not among the most accurate ones. Since the likelihood of any non-observed link can be calculated according to given rules and parameters the most interesting researches in this sub-field are the ones that try to avoid parameters tuning.

The third group of LP algorithms is based on probabilistic Bayesian estimation [68, 69, 70]. Probabilistic models aim at abstracting the underlying structure from the observed network, and then predicting the missing links by using the learned model. Given a target network, the probabilistic model will optimize a built in target function to produce a model composed of a group of parameters that can best fit the observed data of the target network. Then the probability of the existence of a non observed link will be estimated by conditional probability.

The last group of methods employs supervised machine learning techniques. Link prediction through supervised learning algorithms was introduced in [63] where the authors studied the usefulness of graph topological features by testing them on a co-authorship networks dataset. Starting from the obtained scores, and knowing if a link will be present or not in future, a classifier is trained and then used to predict new links.

Since supervised methods are proved to reach better performances both in terms of Area Under ROC (AUCROC) and precision than unsupervised ones, after [63], a wide set of models exploiting several different classification strategies have been proposed. In [71] topological features as well as node attributes of dynamic social network are applied to a covariance matrix adaptation evolution strategy to optimize the prediction. Principal component regression is the algorithm used in [72] to determine the weight of of statistically independent predictor variables. Frequent sub-graph mining based approaches are the ones proposed in [73] and [74], where the first one allows also for the prediction of new nodes. A rank aggregation approach is proposed in [75]. In such work the authors rank the list of unlinked nodes according to some topological measures, then at the new instant time each measure is weighted according to its performance in predicting new links. The learned weights are used in a reinforcing way to forecast subsequent network statuses. In [76] the authors use textual and topological features to predict new citations applying SVM. Finally in [77] tensor factorization is used to select the more predictive topological attributes, whilst in [78] important factors for link prediction are examined and it is provided a general, high-performance framework for the prediction task.

As shown in [59], despite the higher precision supervised approaches as well as maximum likelihood and stochastic approaches can be prohibitively time consuming for networks having over 10000 nodes. In order to reduce the computational complexity several approaches such as [79] make use of clustering and community information to improve the precision of link prediction methods. These analyses suggest that clustering information, no matter the algorithm used, improves link prediction accuracy. Finally, other works [80, 81] show that exploiting time series which model the evolution of continue univariate node characteristics features substantially helps in solving the link prediction task.

In order to efficiently approach the Link Prediction task, it is crucial to identify and calculate a valuable set of graph structural features. When dealing with large-scale graphs that may include millions of vertices and links, one of the challenges is the computationally intensive extraction of such features. Several studies related to link prediction such as [82, 83, 84, 85, 86] try to suggest for each topological structure of a network the best features to use. For example, in [82] is analyzed the relation between network structure and the prediction performance of link prediction algorithm, while in [83] is shown that only small set of features are essential for predicting new edges and that contacts between nodes with high centrality are more predictable than nodes with node centralities.

Moreover, the authors claim that on network with low clustering coefficient LP methods perform poorly, while, as the clustering coefficient grows, the accuracy is drastically improved.

4.1.3 Link-Based Object Ranking

Ranking nodes according to their importance in a network is a classical problem in complex network analysis [87]. Ranking algorithms play a vital and crucial role in complex scenarios such as search engines and social networks. Millions of people use these tools every day and researchers continue the exploration of these algorithms in order to extract significant information. There exist a considerable number of ranking algorithms each one following a different approach and focusing on different aspects of complex networks and reflecting the variety of strategies that are possible to apply. In the past decades several approaches to tackle this task there have been proposed: surely the most popular ones being PageRank, SALSA and some of their derivative [14, 88, 89]. Moreover, some algorithms have been designed to provide multiple rankings in order to better capture the relative importance of node in-going and out-goring connections. The oldest and best-known approach is HITS [47], which provides two rankings (hub and authorities). Another example is topic-sensitive PageRank [90], that can provide an arbitrary number of rankings.

Lastly, some ranking algorithms able to deal with multidimensional networks were proposed: often, however, their outputs strictly relate to the ones computed by PageRank on the monodimensional scenario obtained by collapsing all the network dimensions into a single one. This branch of research has been thoroughly tackled in recent years, as in [91, 92, 93]. Ranking procedures can also be described in order to produce different results depending on specific query involving a subset of the dimensions of the network: the most famous algorithm able to satisfy this request is TOPHITS [94] which, just like HITS, provides two different rankings, hubs and authorities.

4.1.4 Multiplex Networks

Most real life networks are intrinsically multidimensional, and some of their properties may be lost if the different dimensions are not taken into account. In other cases, it is natural to derive multiple dimensions connecting a set of nodes from the available data to the end of analyzing some phenomena.
In order to study this complex scenario a framework that extends the classical graph theory is needed. Reasoning on multidimensional networks seems clear that the usual graph model is not enough to represent all the available information. In their work *"Foundations of Multidimensional Network Analysis"* Berlingerio et al. [95], using a multigraph representation, proposed and evaluated on real datasets a wide spectrum of measurements that are able to capture the interplay among dimensions and to overcome some limits that made the classical monodimensional measures unsuitable in this complex scenario.
Likewise, several researchers have started reasoning on multidimensional/multiplex networks and a wide range of network problems were revised in order to deal with these more complex structures. As example, in [96] is tackled the problem of identify communities in time-dependent multiplex network while in [97] multidimensional hubs are analyzed. Moreover, the multigraph approach discussed in [95] is not the only proposed so far: among all the recently proposed models, the one based on tensor analysis and decomposition is often adopted (as in [98]).
A particular field in which multiplex analysis is increasingly receiving attention is the one offered by the Social Internetworking Scenarios.

Internetworking Scenario

Nowadays, the growing diffusion as well as the increasing sizes of Online Social Networks (OSNs) makes the analysis of *Social Internetworking Scenarios* (SISs) extremely challenging. In a SIS, a user can join multiple OSNs and two users can interact with each other even though they joined different OSNs and did not know each other. While OSNs have been extensively studied in the last

years, the most peculiar aspects of Social Internetworking Scenarios have not been yet investigated, especially from the Social Network Analysis perspective. Many researchers started to collect large amounts of data from OSNs and to apply techniques of classical Social Network Analysis on them. The results they obtained are numerous and extremely interesting: they are based on the intuition that a strong correspondence between the user behavior in an OSN and the structural properties of the corresponding graph is likely to exist. An important aspect to take into account is that nowadays internet users tend to spread their activities among more OSNs and, often, to show a different behavior in each different OSN [99]. As a consequence, different Social Networks could be seen as interconnected thus resulting in a global graph whose structural features are very different from those of each single Social Network *per se*.

Only few commercial attempts to implement Social Internetworking Systems have been proposed so far (i.e. Google Open Social and Friendfeed). Despite the great attention given by scientists towards Social Networks, Social Internetworking Scenarios have been little investigated in the scientific literature also due to their young age. Some papers focus on cross-folksonomies [100, 101], i.e., they analyze the tagging behavior of users in multiple folksonomies and try to relate information about these behaviors. Other ones, such as [102, 103], assume that users can join heterogenous social systems (e.g., a folksonomy and a blogging platform in [103], or Social Networks like Facebook, and social media, like Flickr, in [102]). Their goal is to aggregate user information in such a way as to build a global user profile. Reasoning about such multilevel structures the theory formulated to model and analyze multidimensional networks (where the dimensions are the different OSNs analyzed) represent a very useful resource.

4.2 Topologies: Collective analysis

Extract and analyze meaningful network substructures is likely the most complex and hot topic in network science. In this section we provide a classification of *Community Discovery* algorithms (4.2.1) and relevant works which address two tasks that use network partitions as inputs for complex knowledge extraction processes: *Network Quantification* (4.2.2) and *Services Engagements* (4.2.3).

4.2.1 Community Discovery

The problem of identifying communities in complex networks is very popular among network scientists, as witnessed by an impressive number of valid works in this field. A huge survey by Fortunato [15] explores all the most popular techniques to find communities in complex networks. Traditionally, a community is defined as a dense subgraph, in which the number of edges among the members of the community is significantly higher than the outgoing edges. However, this definition does not cover many real world scenarios [104], and in the years many different solutions started to explore alternative definitions of communities in complex networks [105].
In [105] are identified eight main categories of community discovery: Feature Distance, Internal Density, Bridge Detection, Closeness, Structure Definition, Link Clustering, Meta Clustering and Diffusion:

- *Feature Distance*:
  The algorithms in the *Feature Distance* category usually apply some information theory principles by considering the network as a matrix in which each node is represented as a vector of attributes. Then the matrix is clustered according to these features. One example is Cross Associations [106];

- *Internal Density*:
  In the this category the idea is to partition the network by maximizing the edge density inside the communities. The majority of methods in this category are based on the *modularity* concept, a quality function of a partition proposed by Newman [107], [108]. Modularity scores high values for partitions in which the internal cluster density is higher than the external density. Hundreds of papers have been written about modularity, either using it as a quality function to be optimized, or studying its properties and deficiencies. For instance, two of the issues affecting modularity approaches are the resolution problem and the degeneracy of good solutions [109]. One of the most advanced examples of modularity maximization CD is [96], where the authors use an extension of the modularity formula to cluster multiplex (evolving and/or multirelational) networks. A fast and efficient greedy algorithm, Modularity Unfolding, has been successfully applied to the analysis of huge web graphs of millions of nodes and billions of edges, representing the structure in a subset of the WWW [110];

- *Bridge Detection*:
  The *Bridge Detection* algorithms aim to find similar structure, but with the opposite point of view w.r.t. *Internal Density* approaches: they want to separate the communities by identifying the sparser areas of the network. Examples in this category are [111], [112];

- *Closeness*:
  In the *Closeness* category, Infomap has been proven to be one among the best performing non overlapping algorithms [113]. In the same category of Infomap there is also the Walktrap algorithm [114];

- *Link Clustering*:
  A very important property for community discovery is the ability to return an overlapping coverage, i.e., the possibility of a node to be part of more than one community. This property reflects the common sense intuition that each of us is part of many different communities, including family, work, and probably many hobby-related communities. The above categories

do not consider the possibility of having overlapping communities, and the extensions proposed to do so are usually not universally accepted. The *Link Clustering* category has been created with this specific focus in mind: the community partition is applied on the edges and then the nodes are part of all the communities of their edges (see [115] and [116]);

- *Structure Definition*:
  An overlapping coverage can be achieved also in the *Structure Definition* category, that groups together those algorithms that aim to find a given community structure in the network. An example is the K-CLIQUE percolation algorithm [117];

- *Meta Clustering*:
  Also the *Meta Clustering* category may provide a way to obtain an overlapping coverage, as it is designed to add community features to community discovery algorithms that are part of different categories. An example of such algorithms is HCDF [118]. A similar approach that uses ego networks for community discovery can be found in [119];

- *Diffusion*:
  The *Diffusion* category group together methods that detect communities by spreading labels through the edges of the graph and labeling nodes accordingly to a chosen function of the labels attached to their neighbors. The most famous approach of this family is LABEL PROPAGATION [120] which shows a reasonable good quality on the partition and very low complexity being known as one of the very few quasi-linear solutions to the community discovery problem.

Community Dynamics

Conversely from the static Community Discovery formulation, the problem of finding and tracking communities in an evolutionary context is a relatively novel one. Communities are certainly the structures most affected by changes in network topology: as time goes by the appearance and disappearance of nodes and edges leads to the rising and fall of sub-structures that a static community discovery algorithm is often unable to detect. In order to understand the main directions pursued by researches on community dynamics we can organize all the algorithms proposed so far in four main categories: *life-cycle tracking*, *adaptive community mining*, *online community discovery* and *stable community mining*.

- *Community life-cycle tracking*:
  Strategies that fall in this category are aimed to track the evolution of communities by identifying key actions which regulate their life (i.e., birth, death, merge, split, expand, contract). In [121], authors propose an extended life-cycle model able to track, in an offline fashion, the evolution of communities. This methodology, as well as the one introduced in [122], works on a two-step procedure: (i) the graph is divided in n temporal snapshots and, for each of them, a set of communities is extracted; (ii) for each community an evolutionary chain is built by observing its evolution through temporal adjacent sets (a likelihood measure, e.g., the Jaccard coefficient, is used to identify matches for the same community among different sets).
  In their work, Takaffoli et al. introduced MODEC [123], a framework able to model and detect the evolution of communities obtained at different snapshots in a dynamic social network. The problem of detecting the transition of communities is solved by identifying events that generate the changes of communities across time. Unlike previous approaches [124, 125] the MODEC framework is independent from the static community mining algorithm chosen to partition time-stamped networks.

- *Adaptive Community Mining*:
  A different methodology to discover community in a dynamic scenario, is to design a procedure for which each community structure identified at time t is influenced by the ones

detected at time $t-1$ (avoiding the need to match communities). Adaptive community mining algorithms are proposed in [126, 127, 128, 129]. Although those approaches reduce the complexity of the matching phase, they are still based on a static temporal partition of the complete temporal network.

- *Online Community Discovery*:
  This category of evolutionary approaches is defined by algorithms that do not partition the full temporal annotated graph, but try to build (and maintain) communities in an online fashion following the rising of new nodes and edges. Few works, at the best of our knowledge, has exploited this strategy so far. In [130] a probabilistic approach is proposed to determine dynamic community structure in a social sensing context. The main objective of the introduced IC-DRF model is to dynamically maintain a community partition of moving objects based on trajectory information up to the current time stamp. Given the information used to update the community membership, the approach is suitable only for a specific kind of networked data. Lin et al. [131] proposes an iterative algorithm that, avoiding the classical two-step analysis, extract communities taking care of the topology of the graph at the specific time frame t as well as the historical evolutive patterns of previously computed communities. In [132] Cazabet introduces iLCD an overlapping online approach to community detection.
- *Stable Community Mining*:
  A different way to make use of the temporal information associated to node and edges has been considered in [133], where authors approach the problem of predicting stable community members in an evolutionary and heterogeneous context. Basically, to represent the evolution of the heterogeneous network, two groups of features were extracted: the snapshot-based features and the delta-based features. The snapshot-based ones refer to features that are extracted from the heterogeneous network by taking the elements in the same time window, while the delta-based represents how the snapshot-based features change over time. As results a single community partition of the network is returned.

4.2.2 Network Quantification

The earliest mention of the quantification problem is found in [134], where the task is called *counting*. However, only 10 years later, in 2005, quantification was firstly addressed as a well defined new data mining task [135, 49, 50]. In this series of papers, Foreman proposes several quantification methods and an evaluation measure KLD (Kullback-Leibler divergence). Bella et al. [136] moving from those seminal works, later introduced probabilistic version of Forman's methods.

Quantification has been applied to several domains. For example, [50] uses it to determine the prevalence of support-related issues in incoming telephone calls received at customer support desks, while [137] uses it to estimate the prevalence of response classes in open-ended answers obtained in the context of market research surveys. [138] applies quantification to estimate the distribution of support for different political candidates within blog posts. Differently from all of the above, Xue and Weiss [52] use quantification with the goal of improving the accuracy of classification.

To the best of our knowledge [51] is the only work addressing the quantification problem in the context of networking data, where the goal is estimating class prevalence among a population of nodes in a network. The authors propose an approach, inspired by Forman's equation, which uses network connectivity information to forecast the distribution of binary labels (identified in the following as $+$ and -) for a subset of unlabeled nodes. For each vertex i of the test set (i.e., all the unlabeled nodes of the network) they calculate:

$$p(+) = \frac{p(i) - p(i|-)}{p(i|+) - p(i|-)} \tag{4.1}$$

where $p(i)$ identifies the probability for a generic node v of establishing a link with i, while $p(i|-)$ and $p(i|+)$ denote the conditional probability for v, given its label (- or $+$), to be part of an edge

with i. Once that $p(+)$ and $p(-)$ are computed for all the nodes two steps are performed for the quantification: (i) *Cleaning*: all the computed scores that do not belong to $[0, 1]$ are discarded; (ii) *Class Frequency Estimation*: for each class is returned as frequency estimation the *median* of the cleaned values.

The major problem of this approach is due to the choice of the median as frequency indicator: there is no assurance that the estimation provided for each class will return values that sum to 100%. In their experiments (concerning only datasets with a binary class label), the authors overcome such issue computing $p(+)$ only on a single class and defining the estimate for the second one as its complementary. Obviously, the results obtained by this method varies w.r.t. the initial choice of the class for which computing the median of the distribution. Moreover, due to this choice the applicability of their method is restricted only on a binary class scenario.

A straightforward solution to the quantification problem on networks could be adopting a *sampling* strategy; unfortunately, it does not capture the possible distribution drift. In literature, many works have been proposed to understand the way to choose qualified samples applicable to a hidden population. Unfortunately, these methods do not consider any information about the network structure. Usually, the approaches based on sampling have the form of chain referral sampling [139, 140]. However, the choice on how drawing initial random sample is still a key unsolved problem [141, 142]. Some studies focus on respondent driven sampling [143] for sampling design and population inference in social networks. The process exploits the social structure to expand the initial sample and reduce its independence on it. Moreover, in [144] I proposed a variant of decision trees optimized to perform quantification instead of classification.

4.2.3 Social Engagement

As discussed in 3.2, social engagement has become a major theme of research for several fields and with different purposes. In recent years thanks to the fertile ground provided by social media like Facebook and Twitter, many effort has been devoted to the study of how to predict users' future activities based on their past social behavior. As an example, to this extent in [145] experiments on the chinese social media Renren are conducted using a Social Customer Relationship Management (Social CRM) model able to obtain superior performance when compared with traditional supervised learning methods. Moreover, other works focus in particular on the prediction of churn, i.e., the loss of customers as in [146] where the authors defines the churn probability of a customer as a function of its local influence with immediate social circle, and the churn probability of the entire social circle as obtained from a predictive model. In [147], researchers propose a churn prediction approach based on collective classification, evaluating it using real data provided by the myGamma social networking site. They demonstrate that using such approach and social features derived from the network structure they were able to produce better prediction in comparison to using conventional classification and user profile features only.

A different category of works focus on online advertisement and market targeting on social networks. The work in [148] addresses the problem of online advertising by analyzing user behaviors and social connectivity on online social networks. The authors study the adoption of a paid product by members of the Instant Messenger (IM) network: they first observe that the adoption is more likely if the product has been widely adopted by the individual's friends, and then build predictive models to identify individuals most suited for marketing campaigns. They are able to show that building predictive models for direct marketing and social neighborhood marketing outperforms several widely accepted marketing heuristics. Moreover, in [149] is proposed an evaluation of user's network value in addition to their intrinsic value and its effectiveness in viral marketing, while in [150] is discussed a strategy wherein a carefully chosen set of users is influenced with free distribution of the product and the remaining buyers are exploited for revenue maximization. Finally, in [151] is presented a machine learning approach which combines user behavioral features and social features to estimate the probability of a user click on specific display ad.

4.3 Diffusion of Information

One classic problem in network analysis is understanding how viruses, as well as knowledge, spread within a social context. Modeling diffusion processes on complex networks enables us to tackle problems like preventing epidemic outbreaks [152] or favoring the adoption of new technologies or behaviors by designing an effective word-of-mouth communication strategy. In the last decade, there has been growing interest in the studies of diffusion processes. Two phenomena are tightly linked to the concept of diffusion: the spread of biological [152] or computer [153] viruses, and the spread of ideas and innovation through social networks, the so-called "social contagion" [154, 155]. In both cases, the patterns through which the spreading takes place are determined not just by the properties of the pathogen/idea, but also by the network structures of the population it is affecting.

Some models have been defined to understand the contagion dynamics: the SIR [156], SIS and SIRS [157] models. The idea behind them is that each individual transits between some stages during the life cycle of a disease: i.e., in SIR, from Susceptible (S) to Infected (I), and from Infected to either Recovered (R) or again Susceptible. The availability of Big Data conveying information about human interactions and movements encouraged the production of more accurate data-driven epidemic models. For example, [152] takes into account the space-temporal dimension. Christakis and Fowler studied the role of social prominence in the spread of obesity [158], smoking [159] and happiness [44]. Their results suggest that these health conditions may exhibit some amount of "contagion" in a social sense: although the dynamics of diffusion are different from the biological virus case, they nonetheless can spread through the social network.

A slightly different but equally fascinating problem is the study of innovations and innovators. In his seminal work *"Diffusion of Innovation"* [160], Everett M. Rogers shows how the diffusion of ideas follows a quasi-universal pattern and how the agents which participate in it can be clustered in several well defined categories.

4.4 Temporal Networks

Networks are often used to model dynamic phenomena (i.e., social interactions, biological systems): in order to better describe these realities in which relationships among agents change through time, several works in the last few years have started to lay the foundations of temporal network analysis. Indeed, there is a large number of systems that can, potentially, be modeled as temporal networks. In addition to cellular processes and social communications, large infrastructures (i.e., call graphs and web graphs) posses both network and temporal aspects that make them interesting for temporal network modeling.
The emergence of such theoretical grounds has been clearly highlighted in the book *"Temporal Networks"* [161] where the curators, Holme and Sarämaki, propose an ensemble of works covering different dynamic network analysis methodologies. As done for the multidimensional network scenario, several graph measures were revised and extended in order to deal with dynamic structures: in time dependent networks shortest paths, for instance, will build and change through time, node centrality scores will be different observing the same phenomenon at different stages, diffusion processes will be facilitated or made more difficult by structural and topological perturbations.
To address these problems novel time-aware measures were proposed [162] and, starting from the vast literature on time series analysis, the temporal scale of dynamic processes and burstiness of agent interactions studied [163, 164] in order to extract useful knowledge. Moreover, topological analysis of dynamic networks has involved the extraction of motifs and communities [132, 165, 166, 167] as well as the study of how classical information diffusion models can be fitted in this enriched scenario [168].
Until now, due to the (relatively) limited availability of temporal annotated data, large amount of temporal network studies have been performed on synthetic datasets. Unfortunately there is not yet a commonly agreed set of temporal network characteristics nor a shared temporal network model description that can explain all the possible phenomena able to shape social tissues: for this reason, whenever possible, recent studies has been single phenomenon oriented and data driven. Indeed this approach has advantages as well as drawbacks: if, on one hand, design data driven analysis allows a simplification of the research space (i.e., strong assumption on the data can be made) on the other following this path is quite difficult, or even impossible, to produce models and results that can be considered context independent. For all these reasons, as previously done in static settings, an effort in designing generative models that can output temporal networks with tunable structure is needed. In cases where topological and temporal structures are decoupled, creating a generative model is straightforward. In such scenario, the topology can be generated with any model from the static network literature (modifying for instance the ones seen in 2.3), and then enriched with time series describing node interactions. However, such decoupling does not provides a reliable proxy for the analysis of human sociality development. To overcome such limitations, recently some studies have started to design and adopt time-aware generative models to reproduce and understand dynamics of real world phenomena[169, 170]. Moreover, being able to identify the main characteristics of temporal networks can lead to the design of accurate predictive models aimed at forecasting the future development of interactions and graph sub-structures. Such models, drawn from machine-learning and statistical techniques, would not necessarily attempt to explain why a temporal network is like it is, or to generate contact sequences from scratch: rather, given a contact sequence as well as a series of timestamped graphs, these models aim will be to make predictions on their continuation in the near future.

Chapter 5

Social Network Data

To write it, it took three months;
to conceive it three minutes;
to collect the data in it all my life.
— *F. Scott Fitzgerald*

In this chapter are briefly introduced the data sets used in the 2^{nd} and 3^{rd} part of this thesis. Furthermore we overview the semantics attached to their node and edges, their properties and sources and provide an overall summary. Moreover, in 5.2 are compared different ways to look at networked data: *static* and *dynamic* modeling.

5.1 Social Network Analysis

The advent of OSNs as well as the proliferation of user oriented online services have given rise, during the last decade, to the availability of a heterogeneous set of social data. Almost every information regarding human activities taking place on the internet is nowadays recorded: query logs maintain the history of our searches on the web; email collections describe our communications with friends, family and colleagues; OSNs data capture our interests and relate them to the ones of our contacts; chat and video call logs make possible to describe in detail our sociality as well as purchases from online stores capture our engagement on brands.
Moreover, not only internet-related human activity are recorded and described. Specialized websites collect data regarding offline content such as movies[1], peer reviewed publications[2], political debate, sport related events. Even data regarding terrorism attacks are nowadays described[3], categorized and made available to everyone who is interested in their analysis.
In this data rich scenario network science has found a fertile ground to test its algorithms, develop novel analytical tools and propose solutions to complex problems. Almost all the information we are used to analyze can be modeled using variants of what we call *graph*: due to this explosion of different contexts, carefully described by real data, graph theory is now living a second youth.

The studies introduced in the 2^{nd} and 3^{rd} parts of this thesis will propose analytical tools aimed at extracting knowledge from dataset which describe human "activities". Each algorithm (as well as mining approach) that will be discussed exploit, other than the mere network structure, the semantics offered by the specific category of the analyzed data.

In rest of this section we briefly describe some of the general characteristics of the networks that will be studied in the rest of this thesis (which are summarized in Table 5.1). In particular we will discuss the peculiarities of Social Networks (5.1.1) and Collaboration Networks (5.1.2). To conclude our introduction, in (5.2) we will compare two different classes of networks: static and dynamic ones.

Network	**Nodes**	**Edges**	**Attributes**	**Used In**
FB-LIKE	1 899	113 145	temporal snapshots	8.2
MULTISOCIAL	7 500	97 934	dimensions	6.2 7.1
FB07	19 561	304 392	timestamps	9.1
GOOGLE+	33 381	110 141	node labels	7.2
LAST.FM	75 962	389 639	weekly user's listenings	10.1
WEIBO	8 335 605	49 595 797	timestamps	9.1
SKYPE[4]	xx xxx xxx	xxx xxx xxx	monthly service usage	7.3
CONGRESS	526	14 198	-	7.1
ENRON	5 913	49 058	dimensions, skills	6.3
IMDB_Q	1 440	51 481	node labels	7.2
IMDB_{LP}	14 990	1 106 118	dimensions, temporal snapshots	8.1
IMDB_D	56 542	185 347	-	7.1
DBLP_R	38 942	100 983	dimensions, skills	6.3
DBLP_{LP}	41 836	113 223	dimensions, temporal snapshots	8.1
DBLP_{SLP}	747 700	5 319 654	temporal snapshots	8.2
CORA	4 240	77 842	node labels	7.2
AMAZON	410 236	2 439 437	-	7.1
CG	1 007 567	16 276618	timestamps, geography	8.2 9.1

Table 5.1: Network datasets overview

[1]Internet Movie Database, `www.imdb.com`
[2]DBLP, `www.dblp.org`
[3]Global Terrorism Database, `www.start.umd.edu/gtd/`
[4]The exact number of nodes and edges was removed upon Skype request.

5.1.1 Social Networks

In this category fall all those datasets which are built over samples of Online Social Network users, email exchange logs and telco call graphs. Due to their nature, in these networks two nodes (users, or actors) are connected by a link if among them has occurred an explicit interaction. Social networks can be differentiate by the reciprocity of the connections among users: in some online services (i.e., Facebook, Skype, Last.fm, Foursquare, WEIBO) explicit links have to be considered as fully mutual relationships while in others (i.e. Twitter, Google+) users can specify one-way relations (a user A can *follow* a peer B even if it is not *followed* back). This differentiation leads to the adoption of two different models to represent OSNs: undirected and directed graphs. In our analysis, unless specified otherwise, we will consider the analyzed networks as undirected: most of the obtained results can be, however, generalized for directed scenarios.
Indeed, almost all the networks we will analyze (enumerated in Table 5.1) carry additional semantic informations attached both to nodes and/or edges. In the first set fall Google+, Last.fm and VOIP: the analysis we perform on these networks will aim to characterize the users starting from the topology that surround them and their own features. Conversely, in the second category we can find FB07 (a Facebook regional network built upon users interactions occurred during 2007), FB-Like (a high school social network observed at different period) and Multisocial (a multidimensional/multiplex network built upon a fixed set of users across three different OSNs).
Moreover, among the several types of information that a node/edge can carry a very relevant one is *time*. Social networks describe mutable realities: nodes and edges can appear and disappear. These changes in both local structures and more complex topologies can be captured and exploited during the analytical process. In the 3^{rd} part of this thesis we will analyze some temporal enriched networks, in particular we will work with a temporal annotated sample of Facebook and a call graph of a european mobile carrier.

In order to correctly evaluate the results proposed in the rest of this work, it is very important to understand that both online social networks and telco call graphs can be seen as proxies for human sociality, even if they provide different views of it. The former builds up an overestimation of the real social connection peoples have: OSNs eliminate the costs that are needed to maintain and nurture a friendship (in terms of time, involvements, travel...) amplifying the perception of their users sociality. Conversely, the latter represent an underestimation because: (i) often call data are partial w.r.t. a specific carrier (i.e., we are able to observe only those users whose telco operator is the one we are analyzing) and (ii) call data capture only a subset of the real connections each user have, the strongest one.

5.1.2 Collaboration Networks

A particular typology of human related network is the one described by taking into account as link semantics a group activity. Co-authorship of scientific papers, co-working people (in an office, movie, firm...), athletes playing in the same team are all examples of activities in which the bonding relation among nodes is not restricted to single pairs at once.
These networks can be modeled using simple undirected graphs (due to the intrinsic mutuality of their links semantics) as well as using hyper-graphs. The former model is the most frequently adopted in SNA studies due to its simplicity: this choice however leads to the definition of network structures that are quite dissimilar from the ones describing real social networks. The major differences lie in the higher average clustering coefficient and density: collaboration networks appear, at the eye of the analyst, as built upon chains of cliques of different sizes (i.e., in a co-authorship graph each paper generates a clique involving its authors likewise, in an actor collaboration network, each film generates the full graph of its cast members) while a regular social network is sparse. In the 2^{nd} and 3^{rd} part of this thesis we will use some datasets which fall in this category namely IMDb, DBLP, CoRA and Congress. Moreover, we will use a product-product network built upon the sales of Amazon having the same characteristics of a collaboration one (where the actors are replaced by products and films/papers by carts).

5.2 Static vs. Dynamic Social Networks

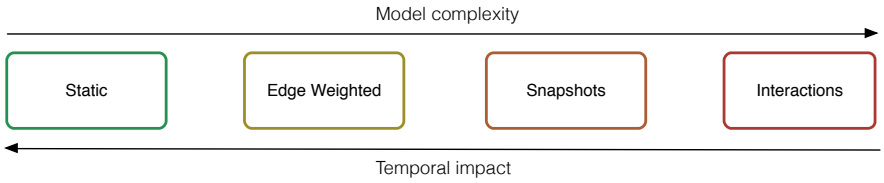


Figure 5.1: From static to dynamic: network representations.

In our daily routines we are used to picture social relations as mutable, dynamic and often rarely persistent bonds. Each one of us is accustomed to live several types of interpersonal relationships: as observed by Dumbar [37], different level of intimacy shape the way we interact with our peers. This complex scenario can be transferred easily on the proxies social network analysis is used to study. Networks inferred from social ties can be used to observe, characterize and forecast different aspects of human activities: varying the purposes of analysis different typologies of information need to be included even when the same phenomenon is modeled. A wide spectrum of SNA works propose solutions for tasks defined on static networks: frequent pattern mining, community discovery, link-based object ranking are only few example of problems that were originally defined on network *"frozen in time"*. The absence of the time variable in classical formulations of analytical tools is due to the fact that they where originally intended for a more general audience: SNA, at its beginning, has moved its steps from *graph theory* and only later has developed its own approach more sensible to the peculiarity of the context it analyze.

In Figure 5.1 are reported four different stages that can be used to describe the gradual introduction of temporal information in a networked context. Moving from the analysis of completely atemporal networks (i.e., a picture at a given, often unspecified, time of a social phenomenon) we can observe how a first solution used to include some temporal information in the model was producing an *aggregate*. In this solution each edge/node carries an attribute which specify its number of occurrences during a pre-specified observation window. This additional information enables, with a small increment of model complexity, for a more detailed analysis and a direct estimate of tie strengths.
As a further improvement of this solution a wide set of works propose the adoption of complete series of network snapshots. Here we can keep track of perturbations introduced in network topology as time goes by looking at an ordered series of network observations. The model complexity slightly increase: indeed, to perform an analysis we need to keep track of more than a single network and to define a mapping functions for nodes and edges. Edge weights and network snapshots suffer however of a common issue: in order to be computed they need as input a desired temporal granularity, a threshold to partition the observed dynamic phenomenon. In order to avoid the definition of such threshold, which is context dependent and often deeply affect the analytical results, in recent years some social network mining tasks where defined to operate on streams. In this approach social networks are seen as composed of temporal ordered *interactions*. Each edge (node) is univocally identified by a time series of punctual observations. Such model offers a complete and fine grained description of the dynamism which regulates the life of a real social network at expenses of a very complex analytical model.
Each one of these stages represent a good fit for a specific set of mining tasks: different formulation for the same problem are often proposed in order to cope with the specific level of temporal information the authors desire to take advantage of. In the following of the thesis we will see how each one of them is able to produce interesting results. In particular we will make use of the static model in 6.1, 6.3, 7.1 and 7.2, of the edge/node temporal aggregation in 6.2, 7.3 and 10.1, of network snapshots in 8.1 and finally of interaction networks in 8.2 and 9.1.

Part II

Frozen in Time: Portrait of a Social Network

Chapter 6

Understanding Local Structures

> If you know the enemy and know yourself you need not fear the results of a hundred battles.
>
> — *Sun Zi*

In this chapter we propose an ensemble of approaches designed to address two well-known structural network problems: Tie Strength evaluation and Node Ranking.
Moreover, here will be discussed novel methodologies able to exploit the additional semantic information expressed by multidimensional networks. The final purpose of the proposed analysis is to show how making leverage on additional knowledge can lead to clearer problem formulation and to higher quality results. As a first step we introduce in 6.1 the framework we adopt to analyze multidimensional networks; afterwards, in 6.2 we show how tie strength can be measured in such semantic rich setting. Finally, in 6.3 the discussion of a ranking algorithm tailored for the Human Resources scenario concludes the chapter.

6.1 Multidimensional Networks[1]

In literature, many analytical measures, both at the local and at the global levels, have been defined in order to describe and analyze properties of standard, monodimensional networks. However networks are often multidimensional: therefore, multidimensional analysis is needed to distinguish among different kinds of interactions or, equivalently, to look at interactions from different perspectives. Analytical measures come under a different light when seen in this setting, since the analysis scenario gets even richer, thanks to the availability of different dimensions to take into account. As a consequence, in this novel setting it becomes indispensable:

- to define a model able to fully represent multidimensional networks and their properties;
- to study how most of the measures defined for classical monodimensional networks can be generalized in order to be applied to multidimensional networks;
- to define new measures, meaningful only in the multidimensional scenario, to capture hidden relationships among different dimensions.

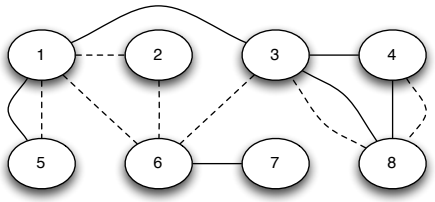


Figure 6.1: Example of a multidimensional network

We choose to use a *multigraph* to model a multidimensional network. For the sake of simplicity, in our model we only consider undirected multigraphs and since we do not consider node labels, hereafter we use *edge-labeled undirected multigraphs*, denoted by a triple $\mathcal{G} = (V, E, L)$ where: V is a set of nodes; L is a set of labels; E is a set of labeled edges, i.e., the set of triples (u, v, d) where $u, v \in V$ are nodes and $d \in L$ is a label. Also, we use the term *dimension* to indicate *label*, and we say that a node *belongs to* or *appears in* a given dimension d if there is at least one edge labeled with d adjacent to it. We also say that an edge *belongs to* or *appears in* a dimension d if its label is d. We assume that given a pair of nodes $u, v \in V$ and a label $d \in L$ only one edge (u, v, d) may exist. Thus, each pair of nodes in $\mathcal{G}$ can be connected by at most $|L|$ possible edges. Hereafter $\mathcal{P}(L)$ denotes the power set of L. An example of multidimensional network is shown in figure 6.1.

6.1.1 Multidimensional network measures

Given a multidimensional network we can extend typical measures defined on traditional monodimensional networks and we can define new measures that are meaningful only in this specific setting. In general, in order to adapt classical measures to the multidimensional setting we need to extend the domain of each function in order to specify the set of dimensions for which they are calculated. Intuitively, when a measure considers a specific set of dimensions, a filter is applied on the multigraph to produce a view of it considering only that specific set, then the measure is calculated over this view.

[1] M. Magnani, A. Monreale, G. Rossetti, and F. Giannotti, "On multidimensional network measures", in SEBD, 2013.

Degree

In order to cope with the multidimensional setting, we can define the degree of a node w.r.t. a single dimension or a set of them. To this end, we have to redefine the domain of the classical degree function by including also the dimensions.

Definition 1 (Degree) *Let $v \in V$ be a node of a network G. The function $Degree : V \times \mathcal{P}(L) \rightarrow \mathbb{N}$ defined as*

$$Degree(v, D) = |\{(u, v, d) \in E \text{ s.t. } u \in V \ \wedge \ d \in D\}| \tag{6.1}$$

computes the number of edges, labeled with one of the dimensions in D, between v and any other node u.

We can consider two particular cases: when $D = L$ we have the degree of the node v within the whole network, while when the set of dimensions D contains only one dimension d we have the degree of v in the dimension d, which is the classical degree of a node in a monodimensional network.

As an example, node 3 in Figure 6.1 has degree centrality $Degree(3, \{d_1, d_2\}) = 5$, indicating its five adjacent edges, while if we focus only on the dashed dimension this reduces to 2.

Neighborhood

In classical graph theory the *degree* of a node refers to the connections of a node in a network: it is defined, in fact, as the number of edges adjacent to a node. In a simple graph, each edge is the sole connection to an adjacent node. In multidimensional networks the degree of a node and the number of nodes adjacent to it is no longer related, since there may be more than one edge between any two nodes. For instance, in Figure 6.1, node 3 has four neighbors and degree equal to 5 (taking into account all the dimensions). In order to capture this difference, we define the following:

Definition 2 (Neighbors) *Let $v \in V$ and $D \subseteq L$ be a node and a set of dimensions of a network $G = (V, E, L)$, respectively. The function $Neighbors : V \times \mathcal{P}(L) \rightarrow \mathbb{N}$ is defined as*

$$Neighbors(v, D) = |NeighborSet(v, D)| \tag{6.2}$$

where $NeighborSet(v, D) = \{u \in V \mid \exists (u, v, d) \in E \ \wedge \ d \in D\}$. This function computes the number of all the nodes directly reachable from node v by edges labeled with dimensions belonging to D.

Note that in the monodimensional case the value of this measure corresponds to the degree. It is easy to see that $Neighbors(v, D) \leq Degree(v)$, but we can also easily say something about the ratio $\frac{Neighbors(v,D)}{Degree(v)}$. When the number of neighbors is small, but each one is connected by many edges to v, we have low values of this ratio, which means that the set of dimensions is somehow redundant w.r.t. the connectivity of that node.

We also define a variant of the Neighbors function, which takes into account only the adjacent nodes that are connected by edges belonging only to a given set of dimensions.

Definition 3 (Neighbors$_{XOR}$) *Let $v \in V$ and $D \subseteq L$ be a node and a set of dimensions of a network $G = (V, E, L)$, respectively. The function $Neighbors_{XOR} : V \times \mathcal{P}(L) \rightarrow \mathbb{N}$ is defined as*

$$Neighbors_{XOR}(v, D) = |\{u \in V | \ \exists d \in D : (u, v, d) \in E \wedge \nexists d' \notin D : (u, v, d') \in E\}| \tag{6.3}$$

As an example, if we consider the dashed dimension in Figure 6.1 the only XOR-neighbor of node 3 is node 6. This indicates how this dimension is fundamental in keeping the two nodes connected — which is not the case for, e.g., nodes 3 and 8, that would be anyway connected through the other dimension.

Multidimensional distance

The main assumption underlying the definition of multidimensional distance is that different edge types are incomparable to each other: if Renzo and Lucia are married and Agnese is Lucia's mother, who is *closer* to Lucia? Our answer is to consider the two relationships as alternative ways of being connected that should not be reduced to a monodimensional concept when distances are computed [171].

For example, in Figure 6.1 we can go from node 3 to node 4 by traversing the continuous edge (3 —— 4), or through two steps along the dashed edge (3 - - - 8 - - - 4) or through a combination of the two (3 —— 8 - - - 4). The number of steps taken in each network determine the multidimensional length of these paths.

Definition 4 (Multidimensional path length) *The multidimensional path length of path p is an array $r_1 + \cdots + r_{|L|}$ where r_i indicates the number of edges traversed in the i^{th} dimension.*

We may also include network switches in this definition – however this better applies to a model emphasizing the separation between the different networks [172], so in this paper we stick to this simpler definition.

The interesting aspect of multidimensional paths is that the choice of not comparing edges of different kinds does not prevent us to identify *shortest* paths containing different edge types. The situation where one multidimensional path is considered shorter than another is formalized using the concept of multidimensional *path dominance*.

Definition 5 (Path dominance) *Let r and s be two multidimensional path lengths. r dominates s iff $\forall l \in [1, |L|]\ r_l \leq s_l \wedge \exists i\ r_l < s_l$.*

As an example, (3 —— 4) and (3 - - - 8 - - - 4) are both shortest paths between nodes 3 and 4, while (3 —— 8 - - - 4) is not: (3 —— 4) is shorter (or *dominates* it) because it involves the same number of steps in the continuous dimension and less steps in the other.

We can thus define the distance between two nodes n_1 and n_2 as follows:

Definition 6 (Multidimensional distance) *Let $ML(n_1, n_2)$ be set of all multidimensional path lengths between nodes n_1 and n_2. The distance between n_1 and n_2 is a set $P \subseteq ML$ such that $\forall p \in P\ \nexists p' \in ML : p'$ dominates p.*

This definition has some attractive properties: it returns all paths that can be shortest under some monotone path evaluation function (e.g., a function assigning a weight to each dimension), does not return any path that cannot be the shortest given some evaluation function, and reduces to traditional path length in the monodimensional case. So, this extension is *tight* (sound and complete) and conservative.

Multidimensional betweenness centrality

Using the concept of multidimensional shortest path introduced in the previous section we can easily compute the betweenness centrality of any node, which characterizes the importance of the node with respect to blocking or favoring information propagation in the network.

In single networks, betweenness centrality is defined as the number of shortest paths passing through a given node. The same definition can be used to characterize betweenness in a multidimensional network by replacing the concept of shortest path with the one introduced in the previous section [173].

It is worth noticing how multidimensional betweenness can capture the role of some nodes that would not emerge in a monodimensional setting. As an example, consider node 8 in Figure 6.1. If we compute its betweenness centrality in the *flattened* network where we do not distinguish between different kinds of edges we can see that no shortest path passes through it, leading to a centrality of 0. On the contrary, the betweenness centrality computed using our measure is positive. For example, a shortest multidimensional path between node 3 and node 4 might pass

through 8 in the case where the dotted dimension is significantly stronger than the continuos one. This comes from the fact that the multidimensional distance between nodes 3 and 4 contains the two alternative paths (3 —— 4) and (3 - - - 8 - - - 4).

Dimension Relevance

One key focus of multidimensional network analysis is on understanding how important a particular dimension is over the others for the connectivity of a node. In the following we introduce the concept of *Dimension Relevance*.

Definition 7 (Dimension Relevance) *Let $v \in V$ and $D \subseteq L$ be a node and a set of dimensions of a network $G = (V, E, L)$, respectively. The function $DR : V \times \mathcal{P}(L) \to [0, 1]$ is defined as*

$$DR(v, D) = \frac{Neighbors(v, D)}{Neighbors(v, L)} \tag{6.4}$$

and computes the ratio between the neighbors of a node v connected by edges belonging to a specific set of dimensions in D and the total number of its neighbors.

Clearly, the set D might also contain only a single dimension d, for which the analyst might want to study the specific role within the network.

However, in a multidimensional setting, this measure may still not cover important information about the connectivity of a node. As an example, this measure does not capture the fact that for a node a dimension could be the only one that allows reaching a subset of its neighbors. To capture this aspect we introduce a variant of this measure.

Definition 8 (Dimension Relevance XOR) *Let $v \in V$ and $D \subseteq L$ be a node and a set of dimensions of a network $G = (V, E, L)$, respectively. $DR_{XOR} : V \times \mathcal{P}(L) \to [0, 1]$ is defined as*

$$DR_{XOR}(v, D) = \frac{Neighbors_{XOR}(v, D)}{Neighbors(v, L)} \tag{6.5}$$

and computes the fraction of neighbors directly reachable from node v following edges belonging only to dimensions D.

In the following, we want to capture the intuitive intermediate value, i.e., the number of neighbors reachable through a dimension, weighted by the number of alternative connections.

Definition 9 (Weighted Dimension Relevance) *Let $v \in V$ and $d \in L$ be a node and a dimension of a network $G = (V, E, L)$, respectively. The function $DR_W : V \times \mathcal{P}(L) \to [0, 1]$, called* Weighted Dimension Relevance*, is defined as*

$$DR_W(v, D) = \frac{\sum_{u \in NeighborSet(v,D)} \frac{n_{uvd}}{n_{uv}}}{Neighbors(v, L)} \tag{6.6}$$

where: n_{uvd} is the number of dimensions which label the edges between two nodes u and v and that belong to D; n_{uv} is the number of dimensions which label the edges between two nodes u and v.

The Weighted Dimension Relevance takes into account both the situations modeled by the previous two definitions. Low values of DR_W for a set of dimensions D are typical of nodes that have a large number of alternative dimensions through which they can reach their neighbors. High values, on the other hand, mean that there are fewer alternatives.

Discussion

In this work we have described a unified model and set of measures to represent and analyze multidimensional networks, i.e., networks with multiple edge types. Some of these measures have already been shown to be useful to identify structural features of a network, e.g., the role of specific dimensions [95], and to study network's hubs [97]. However, they constitute only a first foundational step towards a general extension of complex network mining methods to this multidimensional framework.

6.2 Ties Strength[1]

As we have seen (in 3.1 and 4.1.1) there not exists a common shared formal definition of *"tie strength"* and only few approaches have been proposed with the aim of measuring the intensity of pairwise human interactions. Moreover, social interactions can carry a wide spectrum of different semantics. Moving from this observation, in the following work we propose a measure that, exploiting the multidimensional network analysis tools discussed in 6.1, is able to asses the strength of a tie exploiting the existence of multiple online social links between two individuals. We will discuss, analyzing an Internetworking scenario (as described in 4.1.4), how weak ties often rely on a few commonly available media [174] while strong ones tend to diversify the communication through many different channels [175]. Moreover, we will observe how in presence of strong homophily a high number of different types of relationships tend to form between people pairs.

6.2.1 Multidimensional Formulation

On the vast online world, two individuals can interact and share interests through several social networking platforms. They can be coworkers on LinkedIn, friends on Facebook or Google+, followers/followee on Twitter, they can frequent the same venues on Foursquare, or all of these things together. To express this kind of information we choose to model social contexts as multidimensional networks.
Since strong ties are the ones on which peoples tend to invest more, for example establishing communications through many different channels, it makes sense to model a tie strength measure that exploits the multidimensional nature of online interactions. In order to do so, we define an approaches based on three main features. The first one takes into account the intensity of interaction and the similarity of the nodes in a single dimension:

Definition 10 (Node interaction and similarity)

$$h_d(u,v) = w_d(u,v)\frac{|\Gamma_d(u) \cap \Gamma_d(v)|}{min(|\Gamma_d(u)|, |\Gamma_d(v)|)} \tag{6.7}$$

where w_d is a weight function[2] representing the intensity of the interaction between the nodes in the dimension d, and Γ_d is the set of neighbors of a node in such dimension. In order to capture whether they belong to the same circle of friendships, and whether such circle is prominent for one of them, the intensity of interactions is influenced by the percentage of common neighbors with respect to the more selective node (the one with less friends). The second feature regards the relevance of a dimension for the connectivity of a user: the removal of the links belonging to a dimension should not affect significantly the capacity to reach his real strong connections.

Definition 11 (Connection Redundancy)

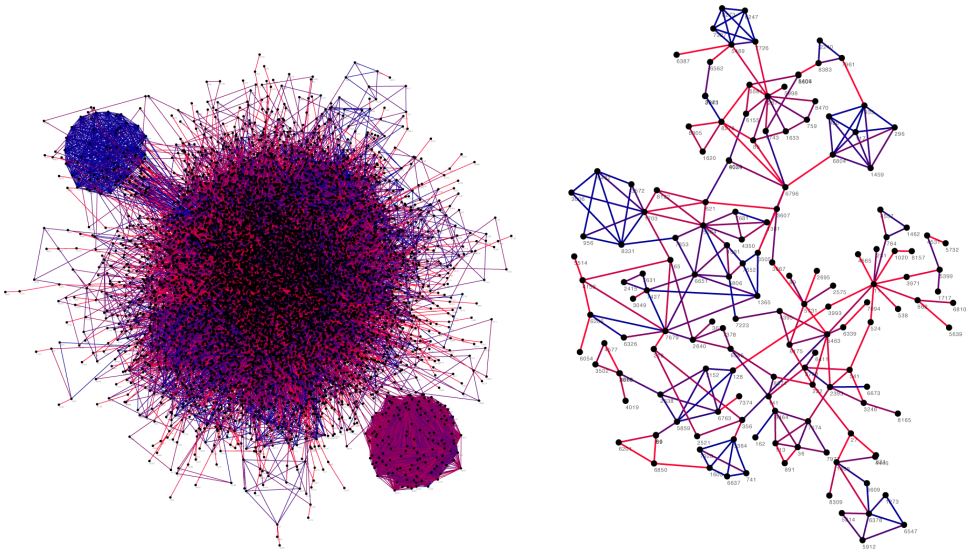
$$\varphi_d(u,v) = (1 - DR(u,d))(1 - DR(v,d)) \tag{6.8}$$

The dimension relevance DR (defined in 7) identifies the fraction of neighbors that become directly unreachable from a node if all the edges in a specified dimension were removed. We give a higher score to the edges that appear in several dimensions, so we are interested in the complement of such values. If two nodes are linked in more than one dimension, the score is raised until a maximum of 1.

We merge these aspects taking into account the multidimensionality of a tie: a greater number of connections on different dimensions is reflected in a greater chance of having a strong tie [176]:

[1] L. Pappalardo, G. Rossetti, and D. Pedreschi, "How well do we know each other? detecting tie strength in multidimensional social networks" in IEEE/ACM ASONAM, 2012

[2] In the rest of the section we will define $w_d(u,v)$ as the number of interactions occurring on d among u and v. However, this function can be furhter specified accordingly to the information available w.r.t. the network analyzed.



(a) (b)

Figure 6.2: *(a)* A global visualization of the network N. *(b)* A portion of the network N. The edge colors ranging from blue to red are used to highlight the computed ties strengths, from strong to weak respectively.

Definition 12 (Multidimensional Tie Strength) *Let $u, v \in V$ be two nodes and L the set of dimension of a multidimensional network $G = (V, E, L)$. The strength function* $\mathbf{str} : V \times V \to \mathbb{R}$ *between two users u, v is defined as:*

$$\mathbf{str}(u, v) = \sum_{d \in D} h_d(u, v)(1 + \varphi_d(u, v)) \tag{6.9}$$

The measure proposed, given its formulation, can be used to estimate the strength of ties even in monodimensional networks: in that scenario the φ_d function assume a value equal to zero and the overall sum became the value of h_d. This scoring function is our final measure of tie strength.

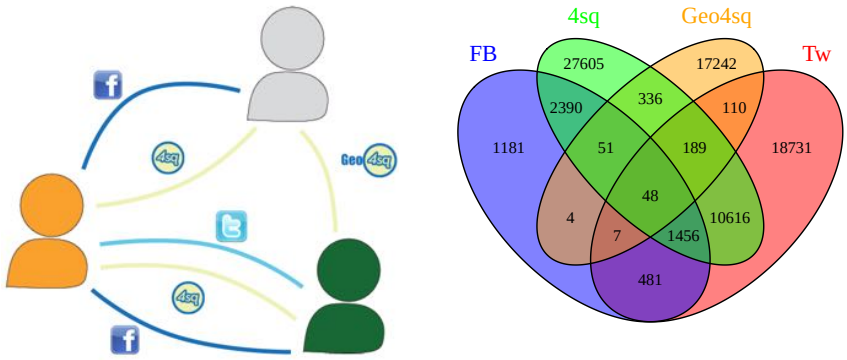
Experiments

We constructed a multidimensional network $G = (V, E, L)$ by collecting friendships existing between the same 7 500 individuals[3] in three online social networks (Foursquare, Twitter and Facebook). Moreover, we inferred a co-occurrence network linking two users if they made a Foursquare checkin in the same venue within a time interval of 15 minutes, during a time span of one month. The number of co-occurrences between two individuals was taken as the weight for the corresponding edge. Figure 6.3(a) presents a schematic example of our 4-dimensional network, whereas Table 6.1 summarizes some characteristics of the multidimensional network and of its dimensions.

[3]All the users considered are geographically located in the city of Osaka (Japan).

Network	**Nodes**	**Edges**	**Weighted**
Foursquare	5 783	42 691	No
GeoFoursquare	4 901	17 987	Yes
Facebook	2 081	5 618	No
Twitter	3 745	31 638	No
Complete Network	7 500	97 934	-

Table 6.1: Basic statistics of the four dimensions as well as for the whole multidimensional network.



(a) (b)

Figure 6.3: (a) A schematization of our 4-dimensional social network. (b) A Venn diagram shoing how the edges of the four dimensions overlap in the multidimensional network.

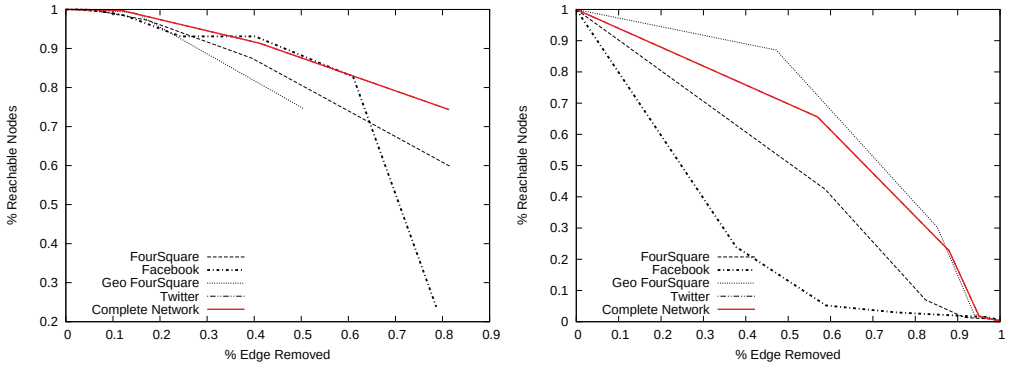
In order to test the meaningfulness of our definition and analyze the structural role of strong and weak links, we calculated the strength measure on G and, using the scores obtained, inferred a weighted network $N = (V, E_N)$, collapsing all the edge between two nodes into one. Figure 6.2 shows a global visualization of N, from which three main clusters clearly emerge, with the one on the left representing people communicating in many different social networking platforms. Furthermore, our measure seems to be consistent with the *"strength of weak ties"* hypothesis [46], with strong tie connecting local communities, and weak ones acting as bridge between them (Figure 6.2). To test more rigorously this aspect, we studied the resilience of N and the individual networks to the removal of either strong and weak links. Since weak ties act as bridges between different communities, we expect that their removal will make the network structure fall apart quickly [40]. Indeed, the deletion of strong ties does not affect considerably the connectivity of the networks, with the 70% of the nodes still reachable in N when almost all the strong arcs were removed (Figure 6.4(a)). Conversely, the removal of weak ties rapidly *"destroys"* the networks, splitting them into several small connected components (Figure 6.4(b)). Our definition is therefore capable to discriminate between intimate circles and the edges acting as bridges between them.

Figure 6.3 shows a Venn diagram representing the number of ties belonging to each possible intersection of the analyzed social dimensions. It clearly shows that there are only 48 bonds appertaining to all the 4 dimensions. Such links represent a sort of *"super strong"* ties, i.e., those having a high probability of being real and intimate friendships.

In order to investigate if the proposed measure correctly assigns the strength values, we analyzed how such scores correlate with three well-known network measure: Jaccard, Adamic-Adar and Edge Betweenness.

Comparing the values assigned by our measure with the corresponding Jaccard coefficient, we want to verify the existence of a correlation between the strength of a tie and the similarity of the individuals involved. We plot the tie strength against the Jaccard coefficient, both for the network N and the single dimensions. As shown in Figure 6.5(a), weak ties tend to have a small Jaccard coefficient, whereas those with higher strength seems more similar. However, there are cases in which a high similarity does not reflect in higher strength. This is because the Jaccard coefficient is defined as the ratio between the common neighbors of the two nodes and all their friends, whereas our measure takes into account the prominence of the circle of friendships.

As done with the Jaccard coefficient, we compare our measure with Adamic-Adar. This measure considers how the mutual neighbors of two nodes are selective in establishing connections: the more selective the friendships are, the more likely the two individuals belong to the same

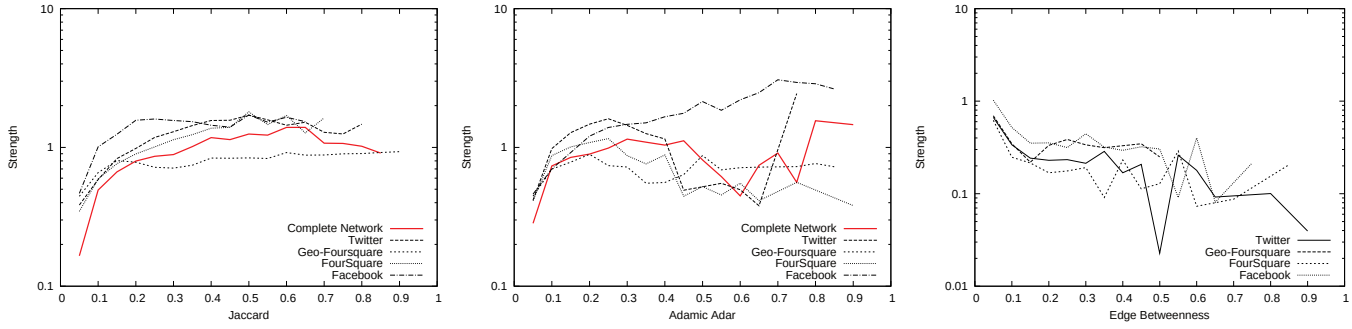


(a) Strong ties removal (b) Weak ties removal

Figure 6.4: Network resilience: *(a)* The stability of the networks to strong link removal. The curves correspond to removing first the high-strength links, moving toward the weaker ones; *(b)* The stability of the networks to weak link removal. The curves correspond to removing first the low-strength links, moving toward the stronger ones.

community. As we can see in Figure 6.5(b), it seems that the strength increases together with the Adamic-Adar score in Facebook, Twitter and the network N. It does not happen with Foursquare, presumably because of the peculiar typology of the service that it offers. Anyway, the trend shown by the figure suggests the following conclusion: two nodes belonging to selective circles of friendships have a greater chance to establish a strong bond.

As seen in 2.2, edge betweenness is a measure of edges centrality. An edge with a high betweenness is likely a bridge between two different communities and, by definition, a weak link. We compare our strength function with this score computed over the single dimensions only. The computation of this measure on the network N is meaningless because, in such network, an edge could establish paths that are not real. As expected, Figure 6.5(c) shows that when the edge betweenness increases, the value of strength seems to decrease.



(a) Jaccard (b) Adamic Adar (c) Edge Betweenness

Figure 6.5: *(a)* Relation between Jaccord coefficient and strength values; *(b)* Relation between Adamic-Adar coefficient and strength values. *(c)* Relation between Edge Betweenness coefficient and strength values.

Discussion

In this work we have introduced a measure of tie strength for multidimensional networks. Supported by a validation on a 4-dimensional SIS, we found that the strength of a tie is strictly related to the number of interactions among the individuals involved. Moreover, it is also related to the number of different contexts in which those connections take place. As future work, we plan to investigate how the information provided by tie strength can be exploited to tackle well-known problems such as link prediction and community discovery.

6.3 Link-Based Object Ranking[1]

Finding talents is one among the most difficult challenges for organizations. Hiring talents means performing better, get more revenue and evolve the business. Where hiring talents is relatively easy, the biggest challenge for organizations today is to find talents they have already hired: finding and creating knowledge is important, but so it is to be able to search and mine knowledge that is already owned. The social networking revolution allowed the creation of tools to evaluate competencies, expertise and skills, like Linkedin. Before social networks, talent management activities were restricted to a marketing oriented approach: to discover talents among their employees, organizations promote internal contests or invest into assessment activities and campaigns. However, during the last few years, organizations started also to put efforts on social talent management, often connected to wider social related initiatives (social CRM, enterprise social networking, etc.). Social talent management is nowadays based on dedicated pages or applications whose aim is to discover interesting professionals. Social networks are also used by organizations to get unofficial information about their employees or candidates, to understand what they do and who they really are.

To understand where and how an employee is positioned on a skill network will enable organizations to find previously hidden sources of knowledge, innovation and know-how. Resumes can provide structured information about studies and working experiences, but they are not useful to understand skills and experiences that do not belong directly to the employee, but to his friends and colleagues. If a candidate does not master a topic but has a strong relationship with somebody who does, then he is an important gateway in that topic, although different relationships allow for different gateway values (i.e., two friends in the same company represents a stronger connection than two friends in competing companies).

If an employee is involved on specific tasks, and she has always been involved only on those tasks, this does not mean she could not have strong competencies and skills on completely different subjects: hobbies, passions, interests can be worth some, sometimes a lot of, value in the *prosumers* age. People are digitally involved throughout their life and there is not a clear separation between personal and professional network. Understanding the position of somebody among different skills and knowledge networks or rankings can let this value emerge. This data richness and hidden knowledge demands for a multidimensional and multi-skill approach to the employee ranking problem: the definition of a ranking algorithm on networks, able to capture the role of different kinds of relations and the importance of different skill sets.

Given a person with a set of skills and a neighborhood of friends in a social network, the skills of the friends to some extent are accessible through that person, and therefore they should be considered when evaluating her. More formally, each node n in a network has some skills S each with a given intensity, and it is connected, through different kinds of edges, to other nodes $(n_2, n_3, \dots)$ with their own skill set $(S_2, S_3, \dots)$. The real value of node n is then defined by a function f that takes as input not only S, but also $S_2, S_3, \dots$, by accounting for the different dimensions connecting n to $n_2, n_3, \dots$. This idea has been proven in the economic field at the macro level: in [177] the author proved that the social return of higher skill levels is higher than the personal return, i.e., higher skilled people make their colleagues to be more valuable as well.

Current ranking algorithms can only provide multiple rankings on monodimensional networks, or simple rankings in multidimensional networks. Hence, we propose an algorithm whose aim is to provide multiple rankings (one for each skill) for nodes in a multidimensional network, a network with multiple types of edges. Our approach is called "you (U) know Because I Know", or UBIK. We test UBIK on real world networks, showing that its ranking is less trivial and more flexible than the current state-of-the-art methods.

[1] M. Coscia, G. Rossetti, D. Pennacchioli, D. Ceccarelli, and F. Giannotti, "You know because i know: A multidimensional network approach to human resources problem", in IEEE/ACM ASONAM, 2013

6.3.1 Network-Based Human Resources

This work aims to tackle the problem of multiple rankings in multidimensional network. Our problem definition is the following:

Definition 13 *Let $G = (V, E, D, S)$ be a multigraph where each node $v \in V$ is connected to its neighbors through multiple edges $e \in E$, each carrying a label $d \in D$; and S be a skill set, such that each node v is labeled with one or more skills $s \in S$, each with a given weight $w \in R^+$. Given a query q containing a set of skills $S_q \subseteq S$ and the importance $r(d) \forall d \in D$, we want to rank nodes accordingly to the weight of each $s \in S_q$ they posses directly or indirectly through their connections.*

The intuition behind our idea is the following. Suppose we have a set of people, each with her own skills and acquaintances, and a task to be performed. In a world without social knowledge interaction, the best way to perform the task is to assign it to the person, or to a set of people (i.e., a team), possessing the highest value of the related skill. However, each person can access to the external knowledge of their acquaintances, thus possibly modifying her skill set value, and therefore the decision of the composition of the team. Each person can also access to the acquaintances' acquaintances skills, but with an increasing cost at each degree of separation, causing at some point the external skill to be useless.

Now, we need to define the social connections, and their different types. We need to formally define the skills carried by each individual as the initial state of the system and how the expertise propagates through social connections.

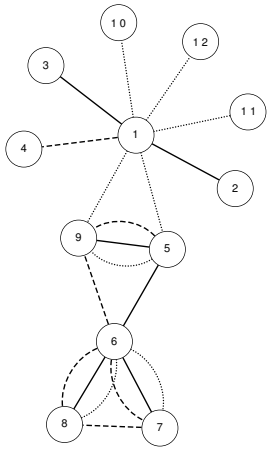
We model our problem with the multidimensional network model as described in 6.1.

However, we need to add several specification to the standard multidimensional network model in order to fit it to our problem definition. First, we need to introduce weighted node labels. In other words, each node $v \in V$ is a collection of couples in the form (s, w) where s is the label and $w \in R^+$ is the value of label s for node v. Therefore, $v = \{(s_1, w_1), (s_2, w_2), \ldots, (s_n, w_n)\}$. The set of node labels describes what in our data are the skills of the node, along with their value. The set of all possible skills is fixed for the network, and we refer to it as S.

Moreover, in 8.1, dimensions are considered distinct but equal. In our case, each dimension can have a different importance: in the real world a friendship tie may be more or less strong than a working collaboration, given the social environment where this tie may play its role. Therefore, each dimension $d \in D$ is represented not only by its label, but also by a value $r(d) \in R^+$, quantifying how much relevant is a relation expressed in dimension d, according to the query requested.

At this point we have all the building bricks of the static part of our model. Next, we define how the knowledge exchange dynamics takes place in the model itself, generating the flow that allows us to rank the nodes in the network. The idea is that each node passes the entire set of its skills to each one of its neighbors. This procedure is similar to the one employed by the classical PageRank formulation, but with a few distinctions. First, our method is not based on random walks, but on the percolation of the various skills without random jumps. Second, the amount of skill value passed to the neighborhood of a node is not equally divided and assigned to each neighbors. The amount of value that node u passes to the neighbor v is proportional to the importance of the dimensions connecting u and v. It is also inversely proportional not only to the degree of the node passing the skill, but also to the degree of the node receiving the skill. Third, the knowledge exchange may be or may be not mutual, i.e., we are not narrowing our model only to directed graphs, but to general graphs.

We use also the range parameter α, commonly used for centrality scores, handling the following situation. If u and v are connected through a node v_2, then the amount of knowledge they exchange is lower than a direct connection, just like the resistance loss in an electric circuit. If x is the amount of value passed by a direct connection, the amount of skills received from nodes ℓ degrees away is corrected as $x^{\frac{1}{\ell \alpha}}$. Traditionally, directly connected nodes should be at zero degrees of separation, because no other *nodes* should be crossed to reach them. For practical purposes, in this paper we assume $\ell = 1$ for neighboring nodes, as we need to cross one *edge* to reach them. The introduction of α is due both to logical and practical reasons. Logically, it makes our model more realistic, as a person is a gateway of her friends' skills, thus she is to some extent also a gateway for her friends'



(a)

Node	Skill-Value
1	$(a, 115), (b, 0), (c, 0), (d, 0)$
2	$(a, 100), (b, 0), (c, 0), (d, 0)$
3	$(a, 100), (b, 0), (c, 0), (d, 0)$
4	$(a, 100), (b, 0), (c, 0), (d, 0)$
5	$(a, 100), (b, 0), (c, 0), (d, 0)$
6	$(a, 0), (b, 0), (c, 0), (d, 80)$
7	$(a, 0), (b, 90), (c, 20), (d, 0)$
8	$(a, 0), (b, 20), (c, 90), (d, 0)$
9	$(a, 100), (b, 0), (c, 0), (d, 0)$
10	$(a, 100), (b, 0), (c, 0), (d, 0)$
11	$(a, 100), (b, 0), (c, 0), (d, 0)$
12	$(a, 100), (b, 0), (c, 0), (d, 0)$

(b)

Figure 6.6: Toy example. (a) The multidimensional network structure (each line style represents a different network dimension; (b) The skill table, recording the values of each skill (a, b, c and d) for each node.

friends' skills, but of a much lower importance. Without the α parameter the skill percolation could potentially continue indefinitely, and the computation may not be able to stop.

In Figure 6.6 we represented a simple toy example. The network structure of social connections is depicted in Figure 6.6(a), while Figure 6.6(b) is the skill table associated with the structure. From the skill table we know that all nodes start with some global skill value (not equal for everybody, as it happens in reality), distributed along four skills (a, b, c and d). The social connections do not have all the same value: solid line has a 50% efficiency in the knowledge transfer, dashed line has a 33% efficiency, while the dotted line has only a 17% efficiency. By looking only at the network structure, node 1 is the most central node. It also has the highest global skill value. According to the closeness centrality, also nodes 5 and 9 are more central than 6. However, our algorithm will propagate skills a, b and c to 6 with the maximum efficiency, while 6 will retain also its unique d skill. At the end of the process, node 6 ends up as the most valuable node in general in the network, while node 1 can only specialize in skill a. With relaxed values for the α parameter (like $\alpha = 1$) node 1 can still get some parts of skill d ($\sqrt{\frac{1}{6}\frac{1}{2}80}$ from node 5 and $\sqrt{\frac{1}{6}\frac{1}{3}80}$ from node 9, that is ~ 4.69017) and even less of b and c. If $\alpha = 3$ then the contributions to node 1 of skills different from a is negligible.

The Data

Our model is describing, according to our hypothesis, how knowledge flows in a face-to-face social environment, following the proven macro level mechanism of the social effect of schooling [177]. However, the data about the face-to-face interactions are usually part of the tacit realm of knowledge. If we cannot find direct or proxy data sources about these interactions, any algorithm solving the problem of evaluating people on the basis of our hypothesis is practically useless. In this section, we present how we use two real-world datasets, adapting them to our model and problem definition, and providing an interpretation of the knowledge that our model can unveil. Of course in both cases we are in front of an approximation. Table 6.2 provides the general statistics about the extracted networks for each dataset.

DBLP[2] is an online bibliography containing information about scientific publications in the field of computer science. Using the data from this dataset, our problem definition may be adapted as follows: we want to evaluate the actual knowledge possessed by scientific authors in different topics

[2] `http://www.dblp.org/db/`

Network	$\lvert V\rvert$	$\lvert E\rvert$	$\lvert D\rvert$	$\lvert S\rvert$	$\lvert D_r\rvert$
DBLP_R	38 942	100 983	16	50	1 187
Enron_N	5 913	49 058	7	8	0

Table 6.2: The statistics of the extracted networks.

and sub-topics, focusing on different branches of their disciplines. Being our aim to rank authors, they should be the nodes of our network. The link is the co-authorship relation: two authors are connected if they have written together a paper. The dimension of the connection should represent the "quality" of their relation, in this case the venue where the publication appeared (we chose 16 top-tier conferences in computer science, including VLDB, SIGKDD, CIKM, ACL, SIGGRAPH and others). The set of skills should describe the expertise of the author, therefore we chose to represent them as the keywords used in their publications. We eliminated stop-words, we applied a stemming algorithm [178] on the remaining words and then we selected the 50 most commonly used keywords in a paper title. The number of times author u used the keyword s is used to evaluate how much the author considers himself an expert over s, i.e., it is used as its w value for s.

The Enron dataset[3] is a collection of publicly available emails exchanged by the employees of the eponymous energy company, distributed after the well known bankruptcy case. We are interested in ranking the employees, that are the nodes of our network. With this network we are able to unveil who are the real knowledge gateways in an organization, by looking at the internal communication even in the absence of more structured social information. Therefore, to apply our algorithm is not necessary for an organization to actually create a social media platform for their employees (or to download information from other social media). We took only the email addresses ending with "@enron.com". We connected two employees if they wrote to each other at least once. Then we used as dimensions the day of the week when the communication took place (ending up with seven dimensions from Monday to Sunday). For the set of skills, we considered the 8 most used keywords in the subject field of the emails (again eliminating stop-words and stemming the remaining words and directly evaluating the relation between an employee and the keyword by the number of times she used the word in an email subject).

6.3.2 The Ubik Algorithm

Given all the necessary premises, we can now discuss the implementation details of our algorithm. We called it Ubik (*"you (U) know Because I Know"*). Ubik requires the following input: a network $G = (V, E, D, S)$ with the characteristics previously discussed; a range parameter α regulating how much information is lost after each degree of separation; and a set specifying, for each $d \in D$, what is the relevance $r(d)$ of d (defined by the analyst accordingly to the ranking aims).

Aim of Ubik is to update $\forall s \in S$ and $\forall u \in V$ the value w, i.e., how much node u possesses of skill s. The pseudocode of Ubik is Algorithm 1. Ubik cycles for each node, using the following master equation:

$$f(u,s) = \sum_{d \in D} \sum_{v \in N(u,d)} \frac{(f(v,s) \times r(d))^{\frac{1}{\ell\alpha}}}{|N(u)| + |N(v)|} \tag{6.10}$$

where $N(u,d)$ is a function returning all the neighbors of u that are reachable through dimension d (if a dimension is not specified, it returns the entire neighborhood). Notice that the contribution of each d is different, corrected with the value $r(d)$, i.e., the relevance of dimension d. Also note that, at the first iteration, $f(v,s)$ (i.e., how much node v possesses of skills s) is equal to just the weight $w_{v,s}$, but at the second iteration it will be updated with the master equation.

One important caveat must be discussed about the ℓ parameter. In the model discussion we said that ℓ represent the degrees of separation of the nodes u and v, exchanging their skill values.

[3] `http://www.cs.cmu.edu/~enron/`

Algorithm 1 The pseudo-code of UBIK.

Require: $G = (V, E, D, S); \alpha \in [0 \dots \infty]$; $r(D)$
Ensure: Node set V with updated skill values.

1: $\ell \leftarrow 1$
2: **while** $\ell < \delta$ **do**
3: **for all** $u \in V$ **do**
4: **for all** $v \in N(u)$ **do**
5: **for all** $s \in S$ **do**
6: $w'_{u,s} \leftarrow w_{u,s} + \sum_{d \in D} \frac{(f(v,s) \times r(d))^{\frac{1}{\ell \alpha}}}{|N(u)|+|N(v)|}$
7: **end for**
8: **end for**
9: **end for**
10: $UPDATE(V, u'(s))$
11: $\ell \leftarrow \ell + 1$
12: **end while**
13: $NORMALIZE(V)$
14: **return** V

Therefore, the exact implementation of our model would require to scan for each u each node of the network, calculate the shortest path between the two, and then update the contribution accordingly to the ℓ value. However, this implementation is inefficient, as it is an equivalent of finding all the shortest paths in the network (that is a cubic problem in terms of the number of nodes, or $O(|V|^3)$ [179]) and then apply our calculation. Instead, Algorithm 1 provides an approximation of the result. The approximation reduces the main loop time complexity as linear in terms of number of edges, usually approximated as $|V| \log |V|$.

We set $\ell = 1$ and we apply the master function to every node and every skill. Then, we increase ℓ by 1 and we apply the master function again, using not the original skill values of nodes, but the ones updated at the first iteration. In this way, all the neighbors of u are passing to u also the skills that they have inherited from their neighbors. We avoid to pass back to u the skills that u itself passed to its neighbors at the previous iteration. At the n-th iteration, the neighbors of u pass to u the skill values obtained by the nodes $n-1$ degrees away.

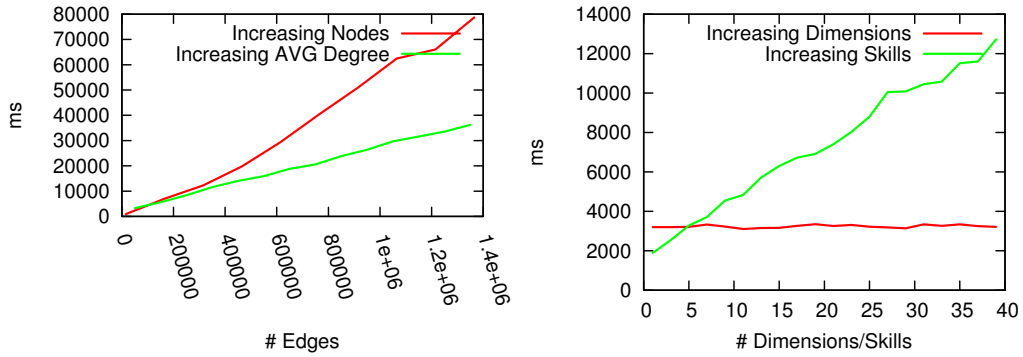
The stop criterion is dependent on the ℓ value. On average, nodes that are beyond three or four degrees of separation cannot influence significantly the skills accessible from one node. Therefore, in Algorithm 1, at step 2 we stop if $\ell \geq \delta$, with $3 \leq \delta \leq 6$, dependent on the application.

When we calculate the new skill values at step 6, we store the result in a temporal variable for each node. Then, we apply the $UPDATE$ function at step 10 to update the value of skill s for node u in the node set V. Each element of the master equation is either fixed (α, D, $r(d)$, $N(u)$ and $N(v)$ are always the same) or it only depends on the previous iteration (ℓ, $u(s)$ and $v(s)$). By forcing this condition, UBIK becomes order-independent: the computed value of each $u(s)$ at a particular iteration is always the same, regardless if u was considered as the first node of the iteration or as the last.

The $NORMALIZE$ function at step 13 scales for each skill the values obtained for each node in the $[0, 1]$ interval. Moreover, it combines all the skill values in a general network ranking. The general value of node u is evaluated by simply extending the $u(s)$ function in the following way: $u(*) = \sum_{s \in S} u(s)$, where $*$ symbolizes the sum of all $s_1, s_2, \dots, s_n \in S$. This function is run at the end of the main UBIK loop and does not add any complexity to it.

Time Complexity

Algorithm 1 has five nested loops. From the inner to the outer, they cycle over: the set of dimensions (the sum at step 6), the set of skills (step 5), the neighbors of a node (step 4), the nodes of the network (step 3) and until the $\ell < \delta$. Since cycling over the nodes and their neighbors (steps 3-4) is equivalent to cycle twice over the edges, the complexity of those two loops is $\mathcal{O}(|E|)$. Steps 5-6 have complexity of $\mathcal{O}(|S|)$, while the outer loop generally terminates after very few



(a) First series. (b) Second series.

Figure 6.7: Running times in milliseconds for different random networks with given number of edges, dimensions or skills.

iterations: in real world networks, usually $\delta = \log |V|$. The final estimate for the time complexity is then $\mathcal{O}(\log |V| \times |E| \times |S|)$. We also report that usually for real world networks, the number of skills and dimensions (both in the order of 10^1 or 10^2) is usually much lower than the number of edges (usually ranging from 10^5 to 10^8 and more), making the average case complexity in the order of $\Theta(\log |V| \times |E|)$.

Experiments

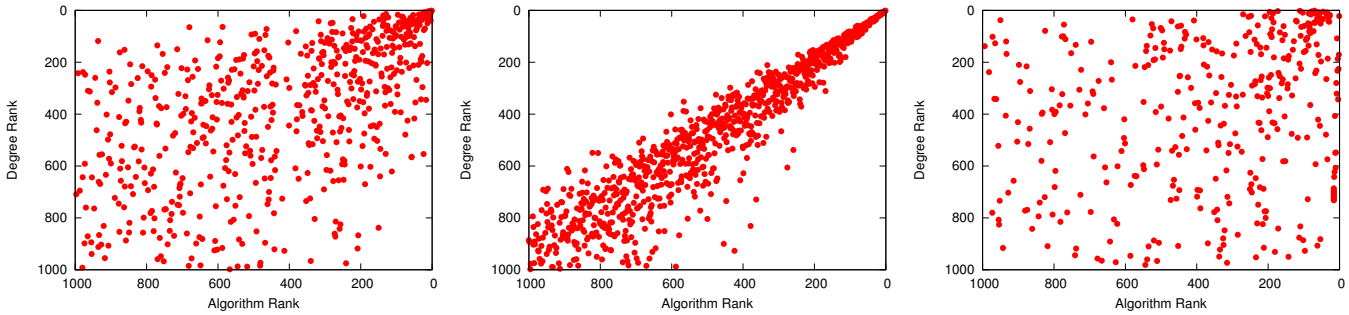
We tested our Java implementation of UBIK[4], on a Dual Core Intel i7 64 bits @ 2.8 GHz, 8 GB of RAM and a kernel Linux 3.2.0-23-generic, using as virtual machine the Java OpenJDK version 1.6.0_24. Our implementation took on average 22 seconds on DBLP and less than 2 seconds on Enron. Our networks are small in scale, thus we created some benchmark networks to show how UBIK scales in terms of number of nodes, average degree, dimensions and skills. The results are depicted in Figures 8.4(a) and 8.4(b).

First, we fixed the number of dimensions and of skills at 5. Then for the "Increasing Nodes" series we fixed the average degree at 3 and we increased the number of nodes; while for the "Increased AVG Degree" series we fixed the number of nodes at 50k and we increased the average degree of the nodes. Both techniques increase the number of edges: UBIK is able to scale linearly in this dimension. UBIK is able to analyze a network with 455k nodes and 1.3M total edges over all dimensions in less than a minute and a half, or with 50k nodes and the same number of edges in less than 40 seconds. The difference in the linear slope between the two is given by the increasing of both nodes and edges. We can conclude that UBIK is scalable and applicable to large scale networks, as it is linear on the number of edges. In Figure 8.4(b) we fixed the number of nodes at 25,000 and the average degree at 3. Then for the "Increasing Dimensions" series we increased the number of dimensions from 1 to 40, while for the series "Increasing Skills" we increased the number of skills from 1 to 40. Our implementation, paying a preprocessing phase, is independent from the number of dimensions, the runtime increases linearly with the number of skills[5].

We now proceed to evaluate the results of UBIK, comparing its results with some of the state-of-the-art node rank approaches and presenting some knowledge extraction examples from real world networks.

[4]Freely available with our test datasets at `http://www.michelecoscia.com/?page_id=480`

[5]To assure repeatability, also the random network generator is provided at the same page of the algorithm and the networks



(a) UBIK. (b) PAGERANK. (c) TOPHISTS.

Figure 6.8: The q-q plots of various ranking algorithms against the ranking obtained ordering the nodes by degree.

Comparison with other methods

We compare the rankings provided by UBIK with some state-of-the-art algorithms. The algorithms used for comparison are the Personalized PAGERANK [90] and TOPHISTS, a tensor eigenvector-based approach to ranking. Personalized PAGERANK is implemented in the R statistical software, TOPHISTS is part of the Tensor Toolbox for MatLab [94], freely available for download[6]. For our comparison, we used the DBLP network.

We used UBIK without giving to any dimension any particular value of $r(d)$ and we took the global ranking of the nodes without selecting any particular skill. In this way, the comparison with PAGERANK and TOPHISTS is fair, because we are evaluating the general rank of our nodes without using anything else than the network structure, that both the Personalized PAGERANK and TOPHISTS can handle. Also, we set $\alpha = 2$ and $\delta = 6$.

The task of confronting different ranking methods is not easy, as it is not explicit why a ranked list is better than a different one on absolute terms. However, there are several properties that we would like to have in the results of a ranking algorithm. We evaluate the results of the algorithm based on quantitative tests on the following properties:

1. Ranking results should not be trivial: if the results are highly correlated with a trivial ranking method, then the algorithm is not telling us something interesting.

2. Ranking results should not be trivially boosted: if there is a simple mechanism to increase one's rank, the flaw of the algorithm makes its results less important.

3. Given a gold standard calculated in an independent way, a ranking algorithm is good if it is quantitatively similar, to some extend, to the gold standard.

Let us start from the first element of the list. One of the most trivial criterion for ranking nodes in a network is to check their degree: the more edges are connected to a node, the more important it is. Of course, this ranking method is not optimal, as it takes only an artificial creation of many edges centered on a node to obtain the maximum rank (as it happens in the World Wide Web). Therefore, the most similar to the degree ranking are the results of the algorithm, the less interesting they are. This is only the first test to be satisfied, but it is necessary to satisfy also the other two. For example, a ranking method that uses the inverse of the degree to rank node will pass this test, as it anti-correlates with the trivial degree ranking method, but it will not satisfy the other two conditions.

Figure 6.8 depicts the q-q plots of UBIK, PAGERANK and TOPHISTS against the degree ranking for the 1,000 nodes with highest degree. Each point $x(i, j)$ of a q-q plot corresponds

[6] `http://www.sandia.gov/~tgkolda/TensorToolbox/index-2.5.html`

R	Degree	PageRank	Ubik
1	Jiawei Han	Jiawei Han	Philip S. Yu
2	Philip S. Yu	Philip S. Yu	Jiawei Han
3	Christos Faloutsos	Christos Faloutsos	Qiang Yang
4	Qiang Yang	Qiang Yang	Hans-Peter Kriegel
5	Divesh Srivastava	Divesh Srivastava	Gerhard Weikum
6	Zheng Chen	Jian Pei	Divesh Srivastava
7	Jian Pei	Zheng Chen	Zheng Chen
8	Raghu Ramakrishnan	Hector Garcia-Molina	Elke A. Rundensteiner
9	Beng Chin Ooi	Beng Chin Ooi	C. Lee Giles
10	Hector Garcia-Molina	Gerhard Weikum	Christos Faloutsos
11	Haixun Wang	Raghu Ramakrishnan	Wei-Ying Ma
12	Wei-Ying Ma	Haixun Wang	Yong Yu
13	Gerhard Weikum	Wei-Ying Ma	Tao Li
14	Michael J. Carey	Michael Stonebraker	Ming-Syan Chen
15	Jeffrey Xu Yu	Rakesh Agrawal	Jian Pei

Table 6.3: The top 15 researchers according to different ranking criteria.

Algorithm	(1)	(2)	(3)	(4)	(5)
Ubik	0.0204	1%	1.2%	4%	5.5%
PageRank	0.0166	0%	0.8%	2%	5.3%
TOPHISTS	0.0857	39%	33.6%	34.8%	37.5%

Table 6.4: The share of high clustering nodes in the top rankings per algorithm.

to some node x. The coordinates of the point (i, j) mean that the node is ranked at the i-th position by the first algorithm (x-axis) and at the j-th position by the second algorithm (y-axis). In Figure 6.8, the y axis is the degree rank, while on the x axis we have Ubik (Figure 6.8a), PageRank (Figure 6.8b) and TOPHISTS (Figure 6.8c). The interpretation of the picture is clear: especially for the 300 highest ranked nodes, having a high degree implies having a high PageRank, while this consideration does not hold for Ubik and TOPHISTS results. We also report the top 15 researchers in Table 6.3 for Ubik and PageRank (TOPHISTS was omitted since the interesting confront of the table is with the PageRank algorithm, as TOPHISTS does not show the rank-degree correlation). Again, we can easily see the correlation between the Degree and the PageRank column. The rank-degree correlation for PageRank is not our finding, as it has already been studied in literature [180]. In practice, the logic of the degree centrality is "The more collaborators a researcher has, the more important he is". Both PageRank and Ubik modify this philosophy in "The more important collaborators a researcher has, the more important he is". However the "important" in PageRank is still a quantitative measure on the number of collaborations, while Ubik uses more qualitative information. For the first criterion, we conclude that Ubik rankings are less trivial than the ones returned by PageRank.

So far we have shown the main defect of PageRank, i.e., it provides a trivial ranking. What is the main defect of TOPHISTS? In literature, it is studied as the Tightly-Knit Community (TKC) effect: being the TOPHISTS rank self-enforced through eigenvector calculation, if we highly clustered nodes in the network, it may happen that all members of this group are ranked high by TOPHISTS, even though the nodes are not particularly central. This is the second point we want to prove: Ubik rankings are not prone to these easily applicable ranking boost strategies.

Table 6.4 reports the share of high clustering nodes for the top k ranked nodes, accordingly to Ubik, PageRank and TOPHISTS. The column (2) reports the percentage of nodes in the top 100 ranked by the algorithm that have a local clustering $k > .1$, the column (3) reports the same

	App.	Search	Stream	Obj.	Analysis	Sys.	User	Model	Network	Context
P. S. Yu	1	3	4	7	4	7	5	6	2	6
J. Han	5	1	3	5	3	6	4	3	3	4
Q. Yang	11	15	1	13	5	14	8	18	7	10
H-P. Kriegel	3	2	5	1	7	12	6	10	16	2
G. Weikum	7	9	15	2	1	17	2	11	5	5
D. Srivastava	18	5	24	21	11	1	12	2	11	7
Z. Chen	17	17	2	20	6	22	1	14	20	17
Rundensteiner	2	7	11	14	9	15	14	1	8	11
C. Lee Giles	14	16	16	16	10	19	7	23	1	14
C. Faloutsos	21	6	34	30	15	28	15	20	30	1

Table 6.5: The top 10 researchers according to the general UBIK ranking and their rank for 10 different skills.

statistic for the top 250 nodes, and so on. The local clustering $k(i)$ of a node i is defined as:

$$k(i) = \frac{2|\{(u,v) \mid u,v \in N(i) \wedge (u,v) \in E\}|}{|N(i)| \times (|N(i)| - 1)} \tag{6.11}$$

(note that we calculate the monodimensional clustering, without specifying a d for $N(i)$). We can see that both UBIK and PAGERANK tend not to return high ranks for nodes with a high local clustering value. On the other hand, in the TOPHISTS ranks 39 nodes out of the most important 100 have high clustering values. Table 6.4 also reports the average local clustering value for the top ranked 100 nodes in column (1) and again this value is $> 4\times$ higher for TOPHISTS. We conclude that both UBIK and PAGERANK are not affected by the TKC, and that UBIK is the only example that is not dependent both on degree and local clustering.

Let us now address the third important feature that a ranking algorithm should have: the comparison with a quantitative gold standard. In scientific publishing, a useful indicator about the quality and the impact of a researcher is quantified using several different indexes. One of them, the h-index, measures both the amount of publications and citations of an author, and it is logically not related to the co-authorship network. A researcher has an h-index of h if he has at least h publications cited at least h times. We use the h-index as our ground truth.

For this comparison we could use a q-q plot, but we need a more quantitative and objective measure than simply looking at the plot. Therefore, we follow [89] and we use a function computing the distance of the points in a q-q plot from the line $y = x$ that represent identical rankings. The distance of point (i, j) from the line $y = x$ is equal to $\frac{|i-j|}{\sqrt{2}}$. Thus, the distance measure D of two rankings r_1 and r_2 is:

$$D(r_1, r_2) = \frac{1}{|V|} \sum_{\forall v \in V} \frac{|r_1(v) - r_2(v)|}{|V|} \tag{6.12}$$

where $r_x(v)$ is the rank of v according to the ranking r_x.

We calculate this function for UBIK, PAGERANK and TOPHISTS against h-index ranking. We obtained the h-index values from an updated webpage who is collecting data from Google Scholar[7]. From that list, we removed the authors that have not published a single paper in the set of conferences with which we have built our multidimensional network, because they are not part of the network structure at all. UBIK's ranking is closer to the ground truth, computed independently from the network structure, provided by the h-index ranking, with a score of 22.43. Therefore, not only UBIK is not affected by the biases of PAGERANK and TOPHISTS, but it also yields results closer to an independent ground truth, as they scored 23.50 and 37.54 respectively. We can now take a look to the actual multi-skill and multidimensional rankings provided by the algorithm,

[7] http://www.cs.ucla.edu/~palsberg/h-number.html

Employee	Weekday Rank	Weekend Rank
Victor Lamadrid	1	34
Jeff Dasovich	2	4
Lisa Jacobson	3	39
Michael Kass	4	2
Joannie Williamson	5	37
Bob Ambrocik	6	68
Chris Germany	7	8
Kay Chapman	8	27
Tana Jones	9	23
Drew Fossum	10	18
Forrest Lane	238	1
Jennifer Blevins	1760	3
Shubh Shrivastava	857	5
Kay Mann	14	6
Scott Neal	50	7
Vince Kaminski	18	9
Rosalee Fleming	21	10

Table 6.6: The top 10 employees according to the UBIK for the weekday and the weekend variants of the ranking.

as the comparison section is over and we can use the features, not handled by PAGERANK nor TOPHISTS.

Multiskill Rankings

We now report some rankings extracted with UBIK. We already saw in Table 6.3 the top 15 researcher setting no particular dimension weight (i.e., $r(d)$ is equal for all $d \in D$). However, now we want to take advantage of the fact that UBIK is able to return different rankings for each skill. In Table 6.5 we report the list of researchers who can master some skills, and their ranking for the other skills. As we can see, no researcher dominates over all the skills, and the different rankings can enlighten us about different leaders in different sectors. We remind that we took only authors of a very specific set of conferences, thus a possible specialist in one or more reported skills may not be part of the rankings because she never published in one of the selected conferences. Also, the skill name is the substantive of the stemmed form, thus it includes all the possible declination of the term (e.g., "Stream", "Streaming", "Streamed", and so on).

UBIK is able to customize the ranking even further, in an additional degree of freedom. Instead of looking at some skills taken separately, we can populate our set of dimension relevance functions with different importance $r(d)$ values for different dimensions. A proper definition of the dimension importance set results in a very specific ranking analysis. To show this feature, we decide to create two different definition classes for the Enron network. We recall that in the Enron network each dimension is the day in the week when the email was sent. In the first variant, we populate our set of rules with a $10\times$ multiplier for the dimensions of Saturday and Sunday; in the second variant we apply the same $10\times$ multiplier, but this time to each of the weekday, and nothing to the weekend days. The choice of the $10\times$ multiplier is *ad hoc*, to represent a query that focuses on the relations established mainly during weekdays (first case) or during weekends (second case).

Table 6.6 reports the top 10 employees according to both criteria (please note that some employees are part of both top 10 and they are not repeated). As we can see, the two rankings are quite distinct. There are three employees very important in both criteria (Jeff Dasovich, Michael Kass and Chris Germany). We also observe one expected phenomenon: the important employees during the weekdays are also somewhat important during the weekends, while the vice versa is not true. It is expected that important employees receive emails during the entire week, while the

Author	ACL	CIKM	ICDE	KDD	GRAPH	VLDB	WWW
D. Marcu	1	3893	3511	2606	2309	1608	3327
Boughanem	2768	1	3892	2966	2643	1946	3658
T. Ichikawa	3070	4549	1	3266	2982	2260	3994
Nakhaeizadeh	2606	4036	3709	1	2484	1783	3507
D. Salesin	2101	3538	3149	2304	1	1300	2980
P. Dubey	3093	4570	4198	3274	2988	1	3999
J. Nieh	2976	4453	4134	3185	2865	2158	1
P. S. Yu	574	32	27	10	860	10	684
J. Han	689	106	60	12	999	29	708
Q. Yang	933	634	967	384	1258	232	930
H-P. Kriegel	1024	517	66	330	1304	146	1476
G. Weikum	1100	472	91	888	1348	51	1355

Table 6.7: The top authors for some conferences (taken singularly) comparative rankings.

communications during the weekend may follow a different logic and promote unexpectedly low ranked employees (maybe because they perform a weekend shift or due to particular emergencies outside office hours).

The most notorious elements of the top management of Enron are not present in either rankings. Kenneth Lay is ranked 617th in weekdays and 224th in weekends, while Jeffrey Skilling is ranked 725th in weekdays and 458th in weekends. Joannie Williamson, who worked as a secretary for both of them[8], is instead present in Table 6.6, and highly ranked. This is an expected result of Ubik, able to unveil who is a knowledge gateway in an organization.

The same multidimensional ranking can be done for the DBLP network. In this case, we are able to spot the collaboration hubs in several different conferences. We applied the $10\times$ multiplier to a collection of conferences. The results for some of our conferences are provided in Table 6.7, where for each conference we record the top ranked author and then we report also his ranking for the other conferences. We can see that Ubik is able to identify specialists who are highly ranked only in one conference. On the other hand, as expected, the top authors in the general ranking score average high rank in all conference, but they are rarely in the top 10 of a specific conference, as their impact is more broad and it spreads over many different venues (the bottom rows of Table 6.7 records the ranking for each single conference for the top 5 general authors in the network taken without any dimension multiplier).

Discussion

In this work we addressed the human resources problem: the ranking of employees according to their skills. We did so following an intuition about the intellectual value of a person: her evaluation should not be based only on her set of skills, but also on her friends' set of skills. We created an algorithm, Ubik, to tackle this problem: Ubik is able to rank nodes in a multidimensional network, with weighted labels referring to their set of skills. We applied Ubik to two real world networks, confronting its output with popular ranking algorithms, showing that our results are less trivial and more significant.

The proposed methodology opens the way to a number of future works. First we can extend Ubik to rank also the relations. Ubik can be applied to several different datasets with different semantics and properties, from Linkedin[9] to evaluate people's expertise; to networks of international organizations, to detect the most important organizations given a set of topics. Moreover, our approach can be easily extended to make use of temporal information attached to nodes and edges in order to design time-aware ranking queries whose results better capture the real values of individual connections.

[8] `http://money.cnn.com/2006/04/03/news/newsmakers/enron_defense/index.htm`
[9] http://www.linkedin.com

Chapter 7

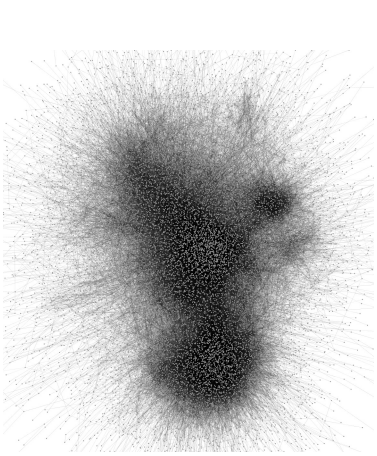
Understanding Topologies

> Every successful individual knows that his or her achievement depends on a community of persons working together.
>
> — *Paul Ryan*

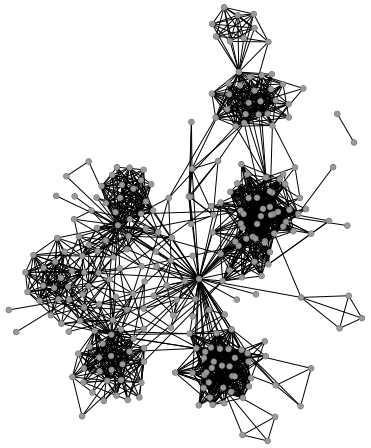
Moving from the *individual/structural* studies proposed in the previous chapters here we discuss one of the most interesting problems related to the analysis of network topologies: Community Discovery. In order to being able to extract and describe communities of social nature, in 7.1 we propose a novel algorithm, DEMON, able to unveil the modular organization of complex networks following a bottom-up approach. We will discuss two variants of the proposed algorithm tailored to produce both overlapping and hierarchical communities.
Moreover, in 7.2 we highlight how, on real world social networks, exploiting the knowledge extracted from communities and ego-networks we are able to gain high performances when approaching the Network Quantification problem. Particularly, we will show that ego-networks and DEMON communities are able to bound homophily of social circles providing valuable insights on the users belonging to them. Later on in 7.3 we will show how this property can be used to classify the level of product engagement of the users of a famous VOIP (Voice Over IP) platform: Skype.

7.1 The Modular Organization of a Network[1]

Complex network analysis has emerged as one of the most exciting domains of data analysis and mining over the last decade. One of its most prolific sub fields is community discovery, or *CD* in short. The concept of a "community" in a (web, social, or informational) network is intuitively understood as a set of entities that have some latent factors in common with each other, and thus play a specific role in the overall function of the complex system [105]. The traditional approach assumes that latent factors drive network connectivity, thus finding sets of nodes with a high edge density among each other and low edge density with the rest of the network effectively detects the functional modules of the network. Community discovery is then a network variant of data clustering, where proximity is replaced with edge connectivity.



(a) A global view of the Facebook graph from 15k users.



(b) The "ego minus ego" network of one Facebook user among the 15k.

Figure 7.1: The real world example of the "local vs global" structure intuition.

The classical problem definition of community discovery assumes that each node plays a single role in the network. This is easily understood by looking at examples of limited size, where it is likely that this assumption is true, as the phenomenon represented is likely to be properly isolated. In this case, the denser areas are easily identifiable by visual inspection. The problem becomes much harder for medium and large scale networks, where many different phenomena are at play at the same time, tangled the one with another. At the global level, very little can be said about the modular structure of most networks: on larger scales the organization of the system becomes simply too complex. As an example, the friendship graph of Facebook includes more than 1.32 billion monthly active users as of August 2014[2]. But even on a tiny fragment of the Facebook friendship graph to assume that there is only a handful of disjoint latent factors at play is naive. In Figure 7.1(a), we depicted the connections among 15 000 nodes, i.e., less than 0.00002% of the total network. Even in this small subset of the network, the friendship dynamics are too interconnected and there is simply no global level organization. In cases like this, the traditional community discovery assumption of a global level disjoint partition tends to return not meaningful communities. The typical aim is to cluster the whole structure and return some huge communities and a long list of small branches (see [105]).

However, as we noted, the structure of cohesive groups of nodes emerge easily considering a *local* fragment of an otherwise big network. The key lies in the fact that in a local view there are few latent factors included and they are usually disjoint one from the other. To use a social

[1] M. Coscia, G. Rossetti, F. Giannotti, and D. Pedreschi, "Uncovering hierarchical and overlapping communities with a local-first approach", ACM TKDD, 2014

[2] http://investor.fb.com/releasedetail.cfm?ReleaseID=861599

metaphor, common sense goes that people are good at identifying the reasons why they know the people they know. In network terms, each node has presumably an incomplete, yet clear, vision of the communities it is part of. Being a bearer of a collection of latent factors, that we represent as labels, he connects to its neighborhood to the nodes bearing the same labels. Then, we can exploit this idea for the CD problem, as illustrated by Figure 7.1(b). Here, we chose one of the 15k nodes from the previous example and extracted what we call its "ego minus ego" network, i.e. its ego network in which the ego node has been removed, together with all its attached edges. Here, it is clear which nodes share which factor, or label, around the ego. Some of these factors are the high school and university friends of the ego, mates from different workplaces and the members of an online community (*We know all these details because the chosen ego is one of the authors of this work*). The ego carries all these labels and knows which subsets of its neighborhood carry one, or more, of these labels too.

Different egos will detect different labels over the same neighbors. The union of all these perspectives, or a hierarchical view in which communities with common labels can be merged together at different aggregation levels, creates an optimal detection of the latent factors of the network. In other words: if node A and node B are considered in the same communities by all the nodes connected to both A and B, then they should be grouped in the same community, because all nodes agree that A and B share the same factors, or labels. If they are considered in the same communities by most or many nodes connected to them, then they are probably part of a higher level super community. This is achieved using a *democratic* bottom-up mining approach: in turn, each node detects the labels attached to the neighbors surrounding it and then all the different perspectives are merged together in an overlapping structure. This overlapping structure can be view as just a flat community coverage, or it can be merged together in a hierarchical fashion.

In the vast CD literature, the general approach does not consider nodes as bearers of different label. The modular structure of a network is usually detected with a (greedy) algorithm, optimizing different quality functions and then returning a set of communities extracted from the global structure. This approach generally ignores the networks latent factors and just operates on the edges of the network without any assumption on why they are distributed as they are. Instead, we propose a change of mentality, in which we make assumptions about the edge distribution and we use these assumptions as drivers of the community discovery process. We propose a simple local-first approach to community discovery in complex networks by letting the latent factors of the organization of a network emerge from local patterns.

Essentially, we adopt a *democratic* approach to the discovery of communities in complex networks. We ask each node to identify the labels it shares with different groups of nodes present in its local view of the network. For this reason, we chose to name our algorithm **D**emocratic **E**stimate of the **M**odular **O**rganization of a **N**etwork, or DEMON in short.
In practice, we extract the ego network of each node and apply a Label Propagation CD algorithm [120] on this structure, ignoring the presence of the ego itself, whose labels will be evaluated by its peers neighbors. We then combine, with equity, the vote of everyone in the network. The result of this combination is a set of (overlapping) modules, our latent factors, detected not by a top-down approach, but by the actors of the network itself. We then either stop the process here, or we consider again a community as a collection of labels, the ones carried by the nodes composing it, connected to other communities by the nodes shared with them, that are the common labels between them. In this way, there is no logical distinction between a community and a node, and therefore we can then reapply the same process and obtain an additional level of the community hierarchy. We repeat the process for each hierarchy level until we collapse the entire network in a set of disconnected communities.

As an alternative example (w.r.t. a social context) to better visualize how our algorithm works, we can imagine to analyze a product network: nodes representing products from a supermarket and edges connecting nodes who share product categories. In the first step, DEMON identifies the micro communities to which a specific product belongs: those are sets of other products sharing, for instance, one or more specific types/categories (fruits, meats, vegetables, gloves, shirts...), while, in the second step, such sets are merged to identify higher-level category definitions (i.e., food, clothing...).

Our democratic algorithm is *incremental*, allowing to recompute the communities only for the newly incoming nodes and edges in an evolving network. Nevertheless, DEMON has also a low theoretical linear time complexity. The main core of our method has also the interesting property of being easily *parallelizable*, since only the ego network information is needed to perform independent computations, and it can be easily combined in a MapReduce framework [181]; although the post-process FlatOverlap procedure is not trivially solvable in a MapReduce framework. The properties of DEMON support its use in massive real world scenarios.

Moreover, in the following we will provide an extensive empirical validation of DEMON. In our experimental setting, we are interested in establishing a link between the communities found by DEMON and the real world knowledge about the labels attached to the nodes. Intuitively, the two should correspond. This is the primary focused objective of DEMON, and we leave other problem definitions to other CD algorithms. First, we test the performance of the algorithms in an established benchmark setting [182], able to generate directed and weighted networks with overlapping communities. Second, we confront the performance with real world networks. In this setting, we make use of a multi label predictor fed with the extracted communities as input, with the aim of correctly classifying the metadata attached to the nodes in real life. Our datasets include the international store Amazon, the database of collaborations in movie industry IMDb, and the register of the activities of the US Congress GovTrack.us. Finally, we provide our latent factor based explanation about why social networks include overlapping communities and why DEMON is good in finding them.

7.1.1 The Demon Algorithm

In this work, our final goal is to find communities in complex networks. Usually, there is some ambiguity connected to the concept of "community" in a complex network [105]. To solve it, we use the latent labels of the nodes as drivers of the community discovery process. In complex and semantically rich settings (such as the modern Web, social networks or other kinds of complex networks), nodes are complex entities carrying multiple attributes that can make them part of different communities for different reasons. In our problem representation, these attributes are represented by the latent labels. We then need to extend the community discovery problem definition to be able to detect them.

Firstly, we define two basic graph operations. The first one is the Ego Network extraction EN. Given a graph $\mathcal{G}$ and a node $v \in V$, $EN(v, \mathcal{G})$ is the subgraph $\mathcal{G}'(V', E')$, where V' is the set containing v and all its neighbors in E, and E' is the subset of E containing all the edges (u, v) where $u, v \in V'$. The second operation is the Graph-Vertex Difference $-g$: $-g(v, \mathcal{G})$ will result in a copy of $\mathcal{G}$ without the vertex v and all edges attached to v. The combination of these two functions yields the $EgoMinusEgo$ function:

$$EgoMinusEgo(v, \mathcal{G}) = -g(v, EN(v, \mathcal{G})) \tag{7.1}$$

Given a graph $\mathcal{G}$ and a node $v \in V$, the set of *local communities* $\mathcal{C}(v)$ of node v is a set of (possibly overlapping) subsets of nodes in $EgoMinusEgo(v, \mathcal{G})$. Each set $C \in \mathcal{C}(v)$ is grouped according to common latent labels. Each node in C shares more common latent labels with other nodes in C, more than with any other node in $C' \in \mathcal{C}(v)$, with $C \neq C'$.

There are two different ways to go from local communities to *global communities*, bringing together an overlapping community coverage with a hierarchical community structure. These two properties have for long been thought as mutually exclusive, but recent approaches proved this assumption wrong [115].

The first is merging the overlapping communities according to the amount of common latent labels they contain, represented by the fact that they share many nodes. In this scenario, we define

Algorithm 2 The pseudo-code of DEMON algorithm.

Require: $\mathcal{G} : (V, E)$; $\mathcal{C} = \emptyset$; μ=0
Ensure: set of overlapping communities $\mathcal{C}$

```
for all v ∈ V do
    e ← EgoMinusEgo(v, G)
    if |e| > μ then
        C(v) ← LabelPropagation(e)
        for all C ∈ C(v) do
            C ← C ∪ v
        end for
    end if
end for
return C
```

the set of communities of a graph $\mathcal{G}$ as:

$$\mathcal{C} = Max(\bigcup_{v \in V} \mathcal{C}(v)) \tag{7.2}$$

where, given a set of sets $\mathcal{S}$, $Max(\mathcal{S})$ denotes the subset of $\mathcal{S}$ formed by its maximal sets only; namely, every set $S \in \mathcal{S}$ such that there is no other set $S' \in \mathcal{S}$ with $S \subset S'$. In other words, by equation (7.2) we generalize from local to global communities by selecting the maximal local communities that cover the entire collection of local communities, each found in the $EgoMinusEgo$ network of each individual node.

In the second approach we recursively apply our logic by seeing the communities as "super nodes" in the network. Just like the nodes, also the communities are collections of latent labels. Therefore, they can be clustered together according to the labels they share. In this approach, the first level of the hierarchy is the set of all $\mathcal{C}(v)$. Then, all communities in $\mathcal{C}(v)$ are collapsed in a single node. Edges are set between the collapsed communities if the two original communities shared at least one node, weighted proportionally to the number of shared nodes. On this new graph structure, the community discovery is applied again. The procedure is repeated recursively until we find a set of disconnected communities.

The Core of the Algorithm

The set of discovered communities $\mathcal{C}$ is initially empty. The external (explicit) loop of DEMON cycles over each individual node, and it is necessary to generate all the possible points of view of the structure and get a complete coverage of the network itself. For each node v, we apply the $EgoMinusEgo(v, \mathcal{G})$ operation, obtaining a graph e. We cannot apply simply the ego network extraction $EN(v, \mathcal{G})$ because the ego node v is directly linked to all nodes $\in EN(v, \mathcal{G})$. This would lead to noise in the subsequent steps of DEMON , since by our definition of local community the nodes would be put in the same community if they are close to each other. Obviously a single node connecting the entire sub-graph will make all nodes very close, even if they are not in the same community. For this reason, we remove the ego from its own ego network. Moreover, we test if the number of nodes within the EgoMinusEgo is greater than a threshold whose value is fixed as input by the user. The μ plays the role of a filter: it can be used to speedup the community extraction process when dealing with massive graphs. For the sake of our analysis in this work we will consider it fixed to 0.

Once we have the e graph, the next step is to compute the communities contained in e. We chose to perform this step by using a community discovery algorithm borrowed from the literature. Our choice fell on the Label Propagation (LP) algorithm [120]. This choice has been made for the following reasons:

1. LP shares with this work the definition of what is a community;

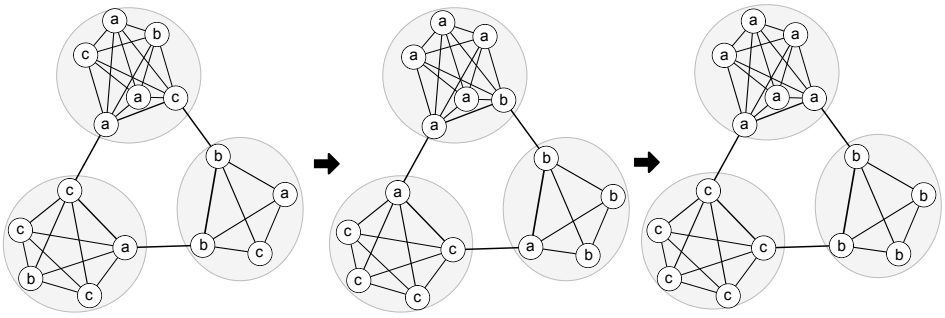


Figure 7.2: A simple simulation of the Label Propagation process for community discovery.

2. LP is known as the least complex algorithm in the literature, reaching a quasi-linear time complexity in terms of nodes;

3. LP will return results of a quality comparable to more complex algorithms [105].

Reason #2 is particularly important, since Step #3 of our pseudo code needs to be performed once for every node of the network. It is unacceptable to spend a super-linear time for each node at this stage, if we want to scale up to millions of nodes and hundreds of millions edges. Given the linear complexity of Step #3, we refer to this as the internal (implicit) loop for finding the local communities.

We briefly describe in more detail the LP algorithm, given its importance in the DEMON algorithm, following the original article [120]. Suppose that a node v has neighbors $v_1, v_2, \ldots, v_k$ and that each neighbor carries a label denoting the community that it belongs to. Then v determines its community based on the labels of its neighbors. A three-step example of this principle is shown in Figure 7.2. The authors assume that each node in the network chooses to join the community to which the maximum number of its neighbors belong. As the labels propagate, densely connected groups of nodes quickly reach a consensus on a unique label. At the end of the propagation process, nodes with the same labels are grouped together as one community. Clearly, a node with an equal maximum number of neighbors in two or more communities can belong to both communities, thus identifying possible overlapping communities. The original algorithm does not handle this situation. For clarity, we report here the procedure of the LP algorithm, that is the expansion of Step #3 of Algorithm 2 and represents our inner loop:

1. Initialize the labels at all nodes in the network. For any given node v, $C_v(0) = v$;

2. Set $t = 1$;

3. Arrange the nodes in the network in a random order and set it to V;

4. For each $v_i \in V$, in the specific order, let $C_{v_i}(t) = f(C_{v_{i1}}(t-1), \ldots, C_{v_{ik}}(t-1))$. f here returns the label occurring with the highest frequency among neighbors and ties are broken uniformly randomly;

5. If every node has a label that the maximum number of their neighbors have, or t hits a maximum number of iterations t_{max} then stop the algorithm. Else, set $t = t + 1$ and go to (3).

At the end of the LP algorithm we reintroduce, in each local community, the ego node v.

The result of Steps #1-7 of Algorithm 2 is a set of local communities $\mathcal{C}$, according to the perspective of all nodes of the network. Please note that there are not repeated communities or communities contained in other communities, as each community has a hash of nodes representing its content. However, these communities are likely to be an incomplete view of the real community coverage of $\mathcal{G}$. Thus, the result of DEMON needs further processing: to merge each local community of $\mathcal{C}$ in order to obtain a proper community coverage. There are two different versions of the function that carries out this task, called Merge function.

Algorithm 3 The pseudo-code of FlatOverlap function.

Require: $\mathcal{C} = Local\ community\ set; \epsilon \in [0..1]$
Ensure: set of global overlapping communities $\mathcal{C}$

for all $C \in \mathcal{C}$ **do**
  for all $I \in \mathcal{C}$ **do**
    if $C \subseteq_\epsilon I$ **then**
      $u = C \cup I$;
      $\mathcal{C} - C; \mathcal{C} - I$;
      $\mathcal{C} = \mathcal{C} \cup u$;
    end if
  end for
end for
return $\mathcal{C}$

Algorithm 4 The pseudo-code of HDemon function.

Require: $\mathcal{G} = (V, E)$, $\mathcal{C} = Local\ community\ set$; $\psi \in [0..1]$
Ensure: set of global overlapping communities $\mathcal{C}$

$l \leftarrow 0$
$\mathcal{C}_0 \leftarrow \mathcal{C}$
while $|\mathcal{C}_l| > |CC(\mathcal{G})|$ **do**
  for all $C_1 \in \mathcal{C}_l$ **do**
    for all $C_2 \in \mathcal{C}_l$ **do**
      $J_{C_1,C_2} = \frac{|C_1 \cap C_2|}{|C_1 \cup C_2|}$
      if $J_{C_1,C_2} \geq \psi$ **then**
        $E \leftarrow E \cup (C_1, C_2, J_{C_1,C_2})$
      end if
    end for
  end for
  $l \leftarrow l + 1$
  $\mathcal{G}_l \leftarrow (V \leftarrow \mathcal{C}_{l-1}, E)$
  $\mathcal{C}_l \leftarrow DEMON(\mathcal{G}_l, \mathcal{C}_l)$
end while
return $\{\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_l\}$

The Flat Overlap Merge

In *FlatOverlap*, two communities C and I are merged if and only if a fraction at most equal to ϵ of the smaller one is not included in the bigger one; in this case, C and I are removed from $\mathcal{C}$ and their union is added to the result set. The ϵ factor is introduced to vary the fraction of common elements provided from each couple of communities: $\epsilon = 0$ ensures that two communities are merged only if one of them is a proper subset of the other, on the other hand with a value of $\epsilon = 1$ even communities that do not share a single node are merged together. Indeed the choice of ϵ impact the overall performances of the merging stage and the number of the communities identified by Demon. The execution time, as well as the number of identified communities, increase as such threshold approach to 0, decrease as it allow a more "shallow" merging policy. This variations are due to the fact that more stringent merging policies are not able to generalize community composition: moreover, allow community merge only in case of proper subset matching determines very low average community size. This procedure is repeated for each community discovered. Returning to the product network example previously introduced, *FlatOverlap* tries to identify higher level communities by merging two sets of nodes if the majority of the products in smallest one also belongs to the bigger one. This iterative procedure aims at identifying groups of products which can describe broader semantic product categories.

The Hierarchical Extension

The HDemon function starts by obtaining the initial set of local communities $\mathcal{C}$. It puts all these communities in a subset of $\mathcal{C}$ and we refer to it as $\mathcal{C}_0$.

For each pair of community discovered HDemon calculates the Jaccard of nodes between the two: if its value is greater than, or equal to, the threshold μ passed as input the merging function connects the two communities, collapsed in a single node, with an edge whose weight is proportional to this amount. At the end of this procedure, we obtain a higher hierarchical view of the original graph $\mathcal{G}$ that we call $\mathcal{G}_1$. The μ parameter acts as the ϵ in the *FlatOverlap* scenario (and as ϵ has its impact on the communities identified as well as on the actual running time of the algorithm).

At this point, HDemon calls again the main core of Demon, this time providing as input not $\mathcal{G}$, but $\mathcal{G}_1$. The resulting local communities of $\mathcal{G}_1$ are put in a separate subset of $\mathcal{C}$ that we call $\mathcal{C}_1$. The procedure is repeated until we find a number of communities in the network equal or lower than the number of connected components of the network ($|CC(\mathcal{G})|$). At this point, we return $\mathcal{C}$, containing all the $\mathcal{C}_0, \mathcal{C}_1, ...\mathcal{C}_i$, representing the hierarchical view of the overlapping communities of $\mathcal{G}$.

Demon Properties

To prove the correctness of the Demon algorithm w.r.t. the problem definition given, we prove by induction some of its properties. It is worthwhile to note that these properties assume that the results of step #3 are constant. The LP algorithm does not satisfy this requirement, i.e., with different random seeds it will return different results. However, here we just want to prove that Demon holds these properties assuming a constant random seed, because this is the crucial feature that enables the incrementality and ability to be parallelized of Demon.

Property 1 (Maximality.) *At the k-th iteration of the outer loop of* Demon*, for all* $k \geq 0$*:*

$$\mathcal{C} = Max(\bigcup_{v=v_1,\ldots,v_k} \mathcal{C}(v)) \tag{7.3}$$

where $v_1, \ldots, v_k$ *are the nodes visited after* k *iterations.*

Property (1) trivially holds for $k = 0$, i.e., at initialization stage. For $k > 0$, assume that the property holds up to $k-1$. Then $\mathcal{C}$ contains the maximal local communities of the subgraph with nodes $v_1, \ldots, v_{k-1}$. By always merging a local community C of node v_k into $\mathcal{C}$ if we find a superset of it in $\mathcal{C}$, we guarantee that C is added to the result only if it is not covered by any pre-existing community, and, if added, any pre-existing community covered by C is removed from $\mathcal{C}$. As a result, after merging all communities in $\mathcal{C}(v_k)$ into $\mathcal{C}$ in Steps #4-6, the latter is the set of maximal communities covering all local communities discovered in $v_1, \ldots, v_k$. Therefore, we can conclude that Demon is a correct and complete implementation of the CD problem stated by equation (7.2). More generally, denoting by $DEMON(\mathcal{G}, \mathcal{C})$ the set of communities $\mathcal{C}'$ obtained by running the Demon algorithm on graph $\mathcal{G}$ starting with the (possibly non-empty) set of communities $\mathcal{C}$, the following properties hold.

Property 2 (Correctness and Completeness.) *If* $DEMON(\mathcal{G}, \mathcal{C}) = \mathcal{C}'$*, where* $\mathcal{G} = (V, E)$*, then*

$$\mathcal{C}' = Max(\mathcal{C} \cup \bigcup_{v \in V} \mathcal{C}(v)) \tag{7.4}$$

In other words, given a pre-existing set of communities $\mathcal{C}$ and a graph $\mathcal{G}$, Demon returns all and only the communities obtained extending $\mathcal{C}$ with the communities found in $\mathcal{G}$, coherently with the definition of communities given in equation (7.2).

Property 3 (Compositionality.) *Consider any partition of a graph* $\mathcal{G}$ *into two subgraphs* $\mathcal{G}_1$*,* $\mathcal{G}_2$ *such that, for any node* v *of* $\mathcal{G}$*, the entire ego network of* v *in* $\mathcal{G}$ *is fully contained either in* $\mathcal{G}_1$ *or* $\mathcal{G}_2$*. Then, given an initial set of communities* $\mathcal{C}$*:*

$$DEMON(\mathcal{G}_1 \cup \mathcal{G}_2, \mathcal{C}) = Max(DEMON(\mathcal{G}_1, \mathcal{C}) \cup DEMON(\mathcal{G}_2, \mathcal{C})) \tag{7.5}$$

This is a consequence of two facts: *i*) each local community $\mathcal{C}(v)$ is correctly computed under the assumption that the subgraphs do not split any ego network, and *ii*) for any two sets of sets $\mathcal{S}_1, \mathcal{S}_2$, $Max(\mathcal{S}_1 \cup \mathcal{S}_2) = Max(Max(\mathcal{S}_1) \cup Max(\mathcal{S}_2))$.

Property 4 (Incrementality.) *Given a graph $\mathcal{G}$, an initial set of communities $\mathcal{C}$ and an incremental update $\Delta\mathcal{G}$ consisting of new nodes and new edges added to $\mathcal{G}$, where $\Delta\mathcal{G}$ contains the entire ego networks of all new nodes and of all the pre-existing nodes reached by new links, then*

$$DEMON(\mathcal{G} \cup \Delta\mathcal{G}, \mathcal{C}) = DEMON(\Delta\mathcal{G}, DEMON(\mathcal{G}, \mathcal{C})) \tag{7.6}$$

This is a consequence of the fact that only the local communities of nodes in $\mathcal{G}$ affected by new links need to be reexamined, so we can run Demon on $\Delta\mathcal{G}$ only, avoiding to run it from scratch on $\mathcal{G} \cup \Delta\mathcal{G}$.

Properties (7.5) and (7.6) have important computational repercussions. The compositionality property entails that the core of Demon algorithm is highly parallelizable, because it can run independently on different fragments of the overall network with a relatively small combination work. Each node of the computer cluster needs to obtain a small fragment of the network, as small as the ego network of one or a few nodes. The Map function is simply the LP algorithm. The incrementality property entails that Demon can efficiently run in a streamed fashion, considering incremental updates of the graph as they arrive in subsequent batches; essentially, incrementality means that it is not necessary to run Demon from scratch as batches of new nodes and new links arrive: the new communities can be found by considering only the ego networks of the nodes affected by the updates (both new nodes and old nodes reached by new links). This does not trivially hold for the Merge function, therefore the actual parallel implementation of Demon is left as future work. However, different and simpler Merge functions can be define to combine the results provided by the core of the algorithm, thus preserving its possibility to scale up in a parallel framework.

Complexity

We now evaluate the time complexity of our approach. Demon core is based on the Label Propagation algorithm, whose complexity is $\mathcal{O}(n+m)$ [120], where n is the number of nodes and m is the number of edges. LP is performed once for each node. Let us assume that we are working with a scale free network, whose degree distribution is $p_k = k^{-\alpha}$. This means that there are $\frac{n}{k^\alpha}$ nodes with degree k. If K is the maximum degree, the complexity would be $\sum_{k=1}^{K}(\frac{n}{k^\alpha} \times (k + \frac{k(k-1)}{2}))$ because for each node of degree k we have an ego network of k nodes and at worst $\frac{k(k-1)}{2}$ edges. This number is very small for the vast majority of nodes, being the degree distribution right skewed, thus many nodes have $k = 1$, thus contributing $\mathcal{O}(0)$ to the complexity; or $k = 2$, thus contributing $\mathcal{O}(1)$. We omit the solution of the sum with the integral and we report that the complexity is then dominated by a single term, ending up to be $\mathcal{O}(nK^{3-\alpha})$. This means that the higher the α exponent, the faster is Demon : with $\alpha = 3$ we have few super-hubs for which we basically check the entire network few times and the rest of nodes add nothing to the complexity; with $\alpha = 2$ we have many high degree nodes and we end up with higher complexity, but still sub-quadratic in term of nodes (as, with $\alpha = 2$, $K << n$).

It has to be noted that this complexity evaluation holds only for the core of the Demon algorithm. The *FlatOverlap* function is more complex, as it has to merge usually thousands of communities. Currently, we have not developed an efficient solution to this problem, that is then quadratic in the number of nodes and dependent on the ϵ parameter.

Experiments

In this section we investigate the performance improvement that Demon provides over the state of the art of community discovery. First, we generate synthetic networks with an established network benchmark generator [182] to evaluate the quality of the community coverage with a

Parameter	Description	Value
N	Number of nodes	1,000
k	Average degree	25
Max k	MaxiMaximum degree	50
μ	Mixing	0.01
Min c	Minimum community size	20
Max c	Maximum community size	50
On	Number of overlapping nodes	500
Om	Number of communities of overlapping nodes	3

Table 7.1: Parameter choice for the benchmark analysis.

known artificial community structure, as a standard robustness check. Then, we switch to three real world networks, namely networks extracted from bill co-sponsorship in the US Congress, the Internet Movie Database and Amazon. We also provide some examples of the insights that it is possible to extract from the flat overlap communities extracted with DEMON as well as from the hierarchical version of the algorithm.

The selected competitors for our assessment are: Hierarchical Link Clustering (HLC) [115], that has been proven able to outperform all the overlapping algorithms, including the k-clique Propagation algorithm by Palla et al [117]; and two overlapping algorithms, the first based on Label Propagation (SLPA [183], [184]) and the second on non-negative matrix factorization (BIGCLAM [185]).

The experiments were performed on a Dual Core Intel i7 64 bits @ 2.8 GHz, equipped with 8 GB of RAM and with a kernel Linux 3.0.0-12-generic (Ubuntu 11.10). The code was developed in Java and it is available for download with the network datasets used[3]. For performances purposes, we mainly refer to the biggest dataset, i.e., Amazon: the core of the algorithm took less than a minute, while the Merge function with decreasing ϵ values can take from one minute to one hour.

Performances on Benchmark Networks

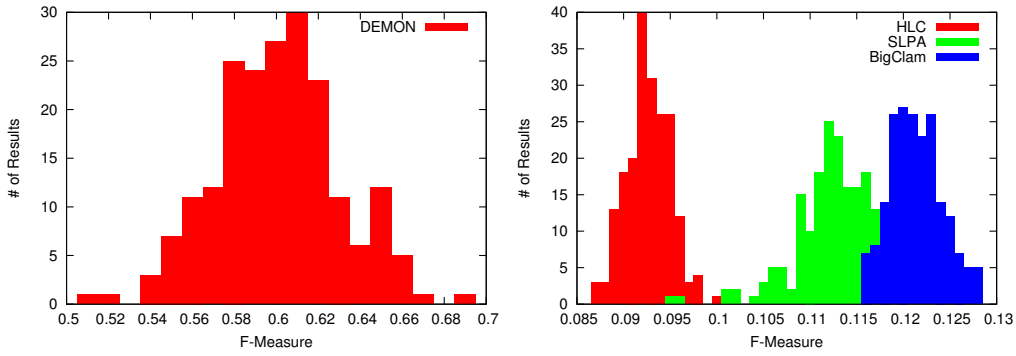
In this section we assess the quality of the community coverage extracted with DEMON using synthetic benchmark networks. The usage of the benchmark networks is useful as we can plug a known community structure and evaluate how well the algorithm is able to uncover it. Of course there are several limitations: as we saw in 4.2.1 there are many different community definitions and benchmark networks can only cover a few. This problem is solved by checking the community discovery quality also in real world networks.

Another problem is that usually benchmark networks are generated in very simple scenarios, i.e., assuming that the network is unweighed, undirected and with non-overlapping community. For this reason, we adopt a benchmark network generator that is able to provide directed, weighted and overlapping networks, that has been introduced in [182].

For each community discovery algorithm we want to test, we generate 200 benchmark networks with a known community structure. The generator developed in [182] requires to specify how many overlapping nodes are in the networks and to how many communities they belong. The parameter choice for the benchmark has been reported in Table 7.1 for experiment repeatability purposes. The generator outputs the network and the communities each node belongs to. Given this information, we can calculate the f-measure between the coverage returned by the community discovery and the real communities of the benchmark network.

Since this is a many-to-many matching problem we adopt a simple strategy of matching the discovered and the original communities. For each discovered community we calculate the f-measure with all the real communities. The community that maximizes the f-measure is the corresponding community and we use that to calculate the average f-measure of the community coverage.

[3]http://kdd.isti.cnr.it/ giulio/demon/



(a) Demon (b) Other Methods

Figure 7.3: The f-measure distribution for the 200 tested benchmark networks.

Figure 7.3 reports for how many benchmark networks (y axis) we obtained a given value of f-measure (x-axis). Particularly, Figure 7.3(a) reports the distribution for Demon , while Figure 7.3(b) reports for all the other methods. We can see from the x axis of the two figures that Demon clearly performs on average significantly better than the other tested algorithms, although it is less stable as its performance is 0.6 ± 0.1 while the other methods' deviation spans from 0.01 to 0.03. In any case, we can conclude that Demon is able to create a much clearer one to one correspondence with the communities in the benchmark networks.

Performances on Real-World Networks

We make use of three networked datasets, representing very different phenomena. We first concentrate on evaluating the quality of a set of communities discovered in these datasets, comparing the results with those of other competing methods in terms of the predictive power of the discovered communities. Since real world data are enriched with annotated information, we measure the ability of each community to predict the semantic information attached with the metadata of the nodes within the community itself. This annotated information is an external explicit information about the latent labels attached to the nodes which drive their connectivity.

Next, we assess the community quality using a global measure of community cohesion, based on the intuition that nodes into the same community should possess similar semantic properties in terms of attached metadata.

Note that we are not able to provide the analytic evaluation for Amazon dataset: for that network HLC algorithm was not able to provide results due to memory consumption problems, while SLPA was not able to conclude in reasonable times.

We tested our algorithms on three real world complex networks extracted from available web services of different domains. A general overview about the statistics of these networks can be found in Table 7.2, where: $|V|$ is the number of nodes, $|E|$ is the number of edges and $\bar{k}$ is the average degree of the network. Congress and IMDb networks are similar to the ones used in [115], generally updating the source dataset with a more recent set of data, and we refer to that paper for a deeper description of them. The networks were generated as follows:

- **Congress**. The network of legislative collaborations between US representatives of the House and the Senate during the 111st US Congress (2009-2011). We downloaded the data about all the bills discussed during the last Congress from GovTrack[4], a web-based service recording the activities of each member of the US Congress. The bills are usually co-sponsored by many politicians. We connect politicians if they have at least 75 co-sponsorships and delete all the connections that are created only by bills with more than 10 co-sponsors. Attached to

[4]http://www.govtrack.us/developers/data.xpd

Network	$\|V\|$	$\|E\|$	$\bar{k}$
Congress	526	14,198	53.98
IMDb	56,542	185,347	6.55
Amazon	410,236	2,439,437	11.89

Table 7.2: Basic statistics of the studied networks.

each bills in the Govtrack data we have also a collection of subjects related to the bill. The set of subjects a politicians frequently worked on is the *qualitative attribute* of this network.

- **IMDb**. We downloaded the entire database of IMDb from their official APIs[5] on August 25th 2011. We focus on actors who star in at least two movies during the years from 2001 to 2010, filtering out television shows, video games, and other performances. We connect actors with at least two movies in which they both appear. This network is weighted according to the number of co-appearances. Our *qualitative attributes* are the user assigned keywords, summarizing the movies each actor has been part of.

- **Amazon**. We downloaded Amazon data from the Stanford Large Network Dataset Collection[6]. In this dataset, frequent co-purchases of products are recorded for the day of May 5th 2003. We transformed the directed network in an undirected version. We also downloaded the metadata information about the products, available in the same repository. Using this metadata, we can define the *qualitative attributes* for each product as its categories.

We first assess DEMON performances using a classical prediction task. We attach the community memberships of a node as known attributes, then its qualitative attributes (real world labels) as target to be predicted; we then feed these attributes to a state-of-the-art label predictor and record its performance. Of course, a node may have one or more known attributes, as we are dealing with overlapping community discoverers; and it may have also one or more unknown attributes, as it can carry many different labels.

For this reason, we need a multi-label classifier, i.e., a learner able to predict multiple target attributes [186]. We chose to use the Binary Relevance Learner. The BRL learns $|L|$ binary classifiers $H_l : X \to \{l, \neg l\}$, one for each different label $l \in L$. It transforms the original data set into $|L|$ data sets D_l that contain all examples of the original data set, labeled as l if the labels of the original example contained l and as $\neg l$ otherwise. It is the same solution used in order to deal with a single-label multi-class problem using a binary classifier. We used the Python implementation provided in the Orange software[7]. For time and memory constraints due to the BRL complexity, for IMDb we used as input only the biggest communities (with more than 15 nodes) and eliminating all nodes that are not part of any of the selected communities.

Multi-label classification requires different metrics than those used in traditional single-label classification. Among the measures that have been proposed in the literature, we use the multi-label version of the standard Precision and Recall measures. Let D_l be our multi-label evaluation data set, consisting of $|D_l|$ multi-label examples $(x_i, Y_i), i = 1..|D_l|, Y_i \subseteq L$. Let H be our BRL multi-label classifier and $Z_i = H(x_i)$ be the set of labels predicted by H for x_i. Then, we can evaluate Precision and Recall of H as:

$$Precision(H, D_l) = \frac{1}{|D_l|} \sum_{i=1}^{|D_l|} \frac{|Y_i \cap Z_i|}{|Z_i|} \tag{7.7}$$

$$Recall(H, D_l) = \frac{1}{|D_l|} \sum_{i=1}^{|D_l|} \frac{|Y_i \cap Z_i|}{|Y_i|} \tag{7.8}$$

[5] http://www.imdb.com/interfaces
[6] http://snap.stanford.edu/data/index.html
[7] http://orange.biolab.si/

Measure	Network	DEMON	HLC	BIGCLAM	SLPA
F-Measure	Congress	**0.21275**	0.14740	0.08987	0.03461
	IMDb	**0.44252**	0.43078	0.38520	0.35431
Accuracy	Congress	**0.10351**	0.08038	0.04420	0.01670
	IMDb	0.34106	**0.38113**	0.33373	0.32011

Table 7.3: The F-Measure scores for Congress and IMDb dataset and each community coverage.

Network	DEMON		HLC		BigClam		SLPA	
	$\lvert\mathcal{C}\rvert$	$\lvert\bar{c}\rvert$	$\lvert\mathcal{C}\rvert$	$\lvert\bar{c}\rvert$	$\lvert\mathcal{C}\rvert$	$\lvert\bar{c}\rvert$	$\lvert\mathcal{C}\rvert$	$\lvert\bar{c}\rvert$
Congress	425	63.3671	1,476	4.5867	99	19.4545	2	263
IMDb	14,004	12.6824	88,119	8.3426	16,411	7.66462	6,717	8.7688

Table 7.4: Statistics of the community set returned by the different algorithms.

We then derive the F-measure from Precision and Recall. We also calculate the multi-label equivalent of Accuracy, that is:

$$Accuracy(H, D_l) = \frac{1}{|D_l|} \sum_{i=1}^{|D_l|} \frac{|Y_i \cap Z_i|}{|Y_i \cup Z_i|} \tag{7.9}$$

These multi-label evaluations are described in [187]. The results are reported in Table 7.3 and show that DEMON comes first for most tests, and second just in one case. We did not test Amazon network as HLC was not able to provide results due to its complexity and further the BRL classifier was not able to scale for the overall number of nodes and labels.

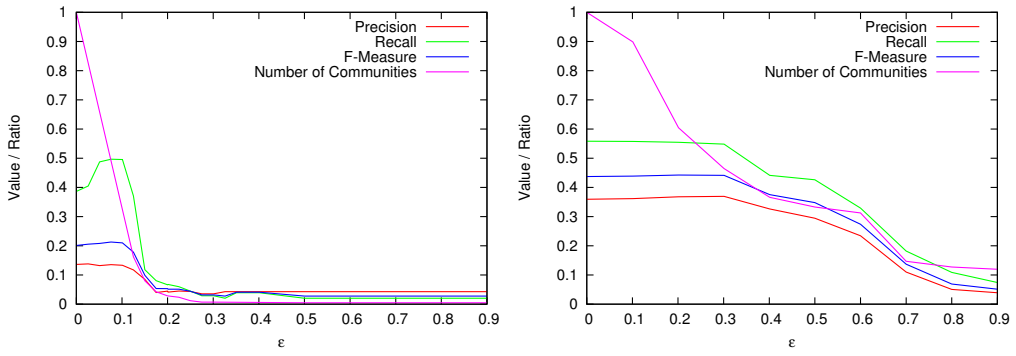
For IMDb dataset, HLC was able to outscore DEMON in Accuracy. However, there is an important distinction to be made about the quantity of the results: if the community discovery returns too many communities, then it is difficult to actually extract useful knowledge from them. We reported in Table 7.4 the basic statistics about the community coverages returned by the algorithms: number of communities ($|\mathcal{C}|$) and average community size ($|\bar{c}|$). For DEMON , we report the statistics of the communities extracted with $\epsilon = 0$. As we can see, DEMON scores are achieved returning 70-80% less communities than HLC.

We report in Table 7.4 the results for $\epsilon = 0$. However, we vary the ϵ threshold and see what happens to the number of communities and to the quality of the results. We report the results in Figure 7.4. We can see that for both Congress and IMDb the Precision, Recall and F-Measure scores remain constant (and actually F-Measure peaks at $\epsilon = 0.076$ and $\epsilon = 0.301$ for Congress and IMDb respectively) before falling for increasing ϵ values, while the relative number of communities dramatically drops. For Congress, we have the maximum F-Measure with only 175 communities; while for IMDb the F-Measure peaks with 6,508 communities (in both cases, less than 50% of the communities at $\epsilon = 0$ and than an order of magnitude of HLC).

A final consideration is needed about the size distribution of the communities detected by DEMON and the other community discovery algorithms. In Figure 7.5 we depicted the community size distribution for DEMON and BIGCLAM for the IMDb network. While BIGCLAM returned more or less the same number of communities of DEMON, we can see that these communities are concentrated in the head of the distribution, i.e., they are on average very small. This is especially true if we consider that these small communities mostly disappear for increasing ϵ thresholds. Communities smaller than a handful nodes are usually less significant and they are often the results of an artifact of the algorithm.

We can conclude that DEMON with a manageable number of medium-sized communities is able to outperform more complex methods and the choice of ϵ can make the difference in obtaining a narrower set of communities with the same (or greater) overall quality.

As presented at the beginning of this section, the networks studied here possess *qualitative attributes*, i.e., a set of annotations attached to each node. Assuming that these qualitative attributes



(a) Congress (b) IMDb

Figure 7.4: Precision, Recall, F-Measure and number of communities for different ϵ values.

correspond to the node's latent factors, we assume that "similar" nodes share more *qualitative attributes* than dissimilar nodes. This procedure is not standard in community discovery results evaluation. Usually authors prefer to use the established measure of Modularity. However, Modularity is strictly (and exclusively) dependent on the graph structure. What we want evaluate is not how a graph measure is maximized, but how good is our community coverage in describing real world knowledge about the clustered entities.

We quantify the matching between a community coverage and the metadata by evaluating how much higher are on average the Jaccard coefficients of the set of *qualitative attributes* for pair of nodes inside the communities over the average of the entire network, or:

$$CQ(P) = \frac{\sum_{(n_1,n_2)\in P} \frac{|QA(n_1)\cap QA(n_2)|}{|QA(n_1)\cup QA(n_2)|}}{\sum_{(n_1,n_2)\in E} \frac{|QA(n_1)\cap QA(n_2)|}{|QA(n_1)\cup QA(n_2)|}}, \tag{7.10}$$

where P is the set of node pairs that share at least one community, $QA(n)$ is the set of qualitative attributes of node n and E is the set of all edges. If $CQ(P) = 1$, then there is no difference between P and the average similarity of nodes, i.e., P is practically random. Lower values implies that we are grouping together dissimilar nodes, higher values are expected for an algorithm able to group together similar nodes.

To calculate the Jaccard coefficient for each pair of the network is computationally prohibitive. Therefore, for IMDb we chose a random set of 400k pairs. Moreover, CQ is biased towards algorithms returning more communities. For this reason, we just collected random communities

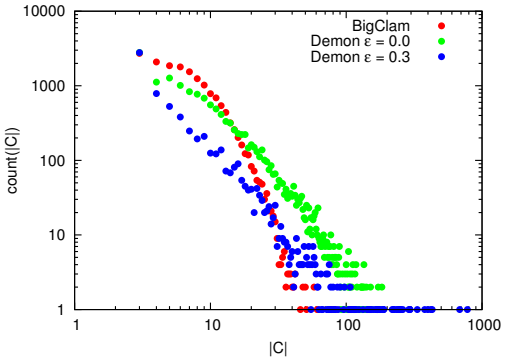


Figure 7.5: The distribution of the community sizes for Demon and BigClam in the Amazon network.

Measure	Network	DEMON	HLC	BIGCLAM	SLPA
CQ	Congress	1.1792	1.1539	**1.2737**	0.9508
	IMDb	**5.6158**	5.1589	0.5954	0.2020
ONMI	Congress	**0.0172**	0.0083	0.0051	0.0127
	IMDb	**0.0536**	0.0429	0.0280	0.0234

Table 7.5: The Community Quality scores for Congress and IMDb dataset and each community coverage.

from the community pool, trying to avoid too much overlap as we want also to maximize the number of nodes considered by CQ (i.e., we try not to consider more than one community per node). We apply this procedure for each algorithm and calculated the CQ value. We repeated this process for 100 iterations and we report in Table 7.5 the average value of the CQ obtained. Also in this case, DEMON was able to outperform all the other algorithms in three out of four cases, ending up second in one case, this time to BIGCLAM instead of HLC.

We also calculated the Overlapping Mutual Information between the set of communities selected as described above and the collection of labels attached to the nodes. Traditional Mutual Information is not defined for a multi-label setting. Some researchers defined a Normalized Mutual Information for this purpose [188]. However, further research [189] pointed out that this version of the Normalized Mutual Information has some drawbacks, namely its unintuitive behavior. For this reason, we used the measure defined in [189] and we refer to it as "ONMI". The results of our experiments are again reported in Table 7.5. This time, we can observe that DEMON is able to outperform all the considered alternatives.

Overlapping Communities

In this section we present a brief case study using the communities extracted for the previously exposed evaluation of DEMON , that uses the *FlatOverlap* function to merge the communities. We focus on the Amazon network. Aim of the section is to DEMONstrate that the overlap between the extracted communities carries meaningful information. By analyzing the overlap, we can have practical applications in the extraction of knowledge from real world scenarios. In the next section we focus instead on the hierarchy of the communities, rather than their overlap.

In the Amazon network to have different communities for each item is very useful. A recommendation system is able to better discern if a user may be interested in a product or not given that he bought something else; however, being part of one community of products does not mean that that particular community describes all aspects of a particular product.

Let us consider, as an example, the case of Jared Diamond's best selling book "Guns, Germs, and Steel: The Fates of Human Societies". Clearly, it is difficult to say that the people interested in

Figure 7.6: A representation of parts of the two communities surrounding our case study in the amazon network.

Rank	Level 0	Level 1
1	Sen. Com. on Comm., Science, and Transport	Sen. Com. on Foreign Rel.
2	Sen. Com. on Foreign Rel.	Sen. Com. on Comm., Science, and Transport
3	Int. scientific coop.	Sen. Spec. Com. on Aging
4	Office of Science and Tech. Policy	Sen. Com. on Env.t and Public Works

Rank	Level 2
1	Sen. Com. on Comm., Science, and Transport
2	Sen. Com. on Indian Affairs
3	Sen. Spec. Com. on Aging
4	Sen. Com. on Health, Edu., Labor, and Pens.

Table 7.6: The top four topics of one community in the Congress network across the hierarchy.

this book are always interested in the same things. Checking the communities to which it belongs, we find two very different big communities (a depiction of the two communities is provided by Figure 7.6). These communities have some sort of overlap, however they can be characterized by looking at the products that appear exclusively in one or in the other. In the first one we find books such as: "Beyond the State: An Introductory Critique", "The Econometrics of Corporate Governance Studies" and "The Transformation of Governance: Public Administration for Twenty-First Century America". This is clearly a community composed mainly by purchases made by the people more interested in the socio-economic aspects of Diamond's book. The second community hosts products such as: "An Introduction to Metaphysics", "Aristophanes' Clouds Translated With Notes and Introduction" and "Being and Dialectic: Metaphysics and Culture". This second community is apparently composed by the purchases of customers more attracted by the underlying philosophical implications of Diamond's study. Products in one community may have something in common, but they are part of two distinct and very well characterized groups, and the ones in one group are not expected to be found in the other.

This is of course one of the many cases. We report as an additional example the two communities around the historical novel "The Name of the Rose" by Umberto Eco: one community is characterized by history related products (such as "Ancestral Passions : The Leakey Family and the Quest for Humankind's Beginnings"), the other by costume fiction (for example the 1932 Dreyer's movie "Vampyr").

Hierarchical Communities

The aim of this section is to DEMONstrate that, besides the overlap, also the hierarchy of the extracted communities carries meaningful information. In this case we focus on the Congress network, as the US Congress has a particular structure that is easy to confront with. In the US, the Congress is divided in two parts: the House and the Senate. Members of the House are not members of the Senate and vice versa. Then, inside both the House and the Senate, there are several subcommittees, each with a different focus. We expect to find members of similar subcommittees in the communities at the lower level of the hierarchy, and just two large communities at the top of the hierarchy: the community of the House and the community of the Senate.

We report in Table 7.6 the first four topics of a particular community across the entire hierarchy. As we can see, at the bottom level this community is clearly a senate community composed by a small group of senators very focused on science and technology. In the intermediate and top level, the community merges with more and more general communities from the senate. At level 1 is still focused on broader social issues, while at level 2 it is basically composed by almost any senate committee. At level 2, we expected to find only two communities. We found, instead, 28 of them. However, these 28 communities are easily split into the two expected groups with little overlap. Since DEMON does not return any community with less than three nodes, it does not create the additional hierarchy level with the two nodes representing the House and the Senate communities.

7.1.2 The Overlap in Social Networks

As we saw in the previous sections and in countless other examples in literature, overlapping communities are ubiquitous in many social and complex networks. We chose as our explanation of the overlap the fact that many different latent factors drive the nodes' connectivity. Nodes are collections of latent labels and they tend to connect with nodes with similar labels. This is the assumption we share with the creators of the BigClam algorithm [185].

This corresponds to the most successful explanation used for overlapping communities: in a network there are different types of relations whose interplay brings together people from different communities (the starting assumption of the HLC algorithm [115]). For example, one person is part of the community of her college mates and also of the sport team she practices. Several of her team-mates may be also college mates, generating an overlap between the two communities. In this section we do not focus on a systematic proof of this theory, that goes beyond the scope of this thesis. We provide, instead, empirical evidences of this theory, along with the proof that Demon is able to correctly detect actual overlap.

Network	Nodes	Edges	Facebook	Twitter	Foursquare
Facebook	2,081	5,618	1	0.57	0.94
Twitter	3,745	31,638	0.32	1	0.85
FourSquare	5,783	42,691	0.34	0.55	1
Total	7,461	79,947	-	-	-

Table 7.7: The statistics of our multidimensional network per dimension.

To do so, we need to add context information to the relationships connecting two people. One of the most important approach to this problem in literature involves the use of multidimensional networks. Community discovery in multidimensional networks is a problem studied in [190] and it requires specialized multidimensional community discovery algorithms. Demon is not a multidimensional community discovery algorithm, so we need to create a special analytic setting.

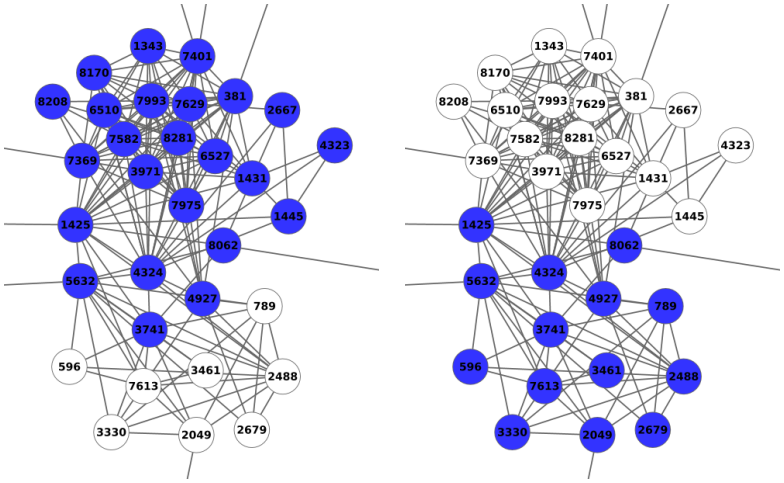
We create a multidimensional network by joining three different social networks: Facebook, Twitter and Foursquare. We were able to crawl the relationships of the same set of users in these three social media websites. Table 7.7 records some statistics about the topology of the three social networks and their aggregate multidimensional network. For each dimension of the network we report the number of nodes and edges appearing in it and the node overlap with the other network dimensions.

We then applied Demon to each dimension of the network separately. The algorithm found overlapping communities for each dimension of the network without information about the edges from the other dimensions. The aim of our analysis is to examine communities with a large overlap in one dimension and verify the community affiliations in the other dimensions of the network of the nodes belonging to these two communities.

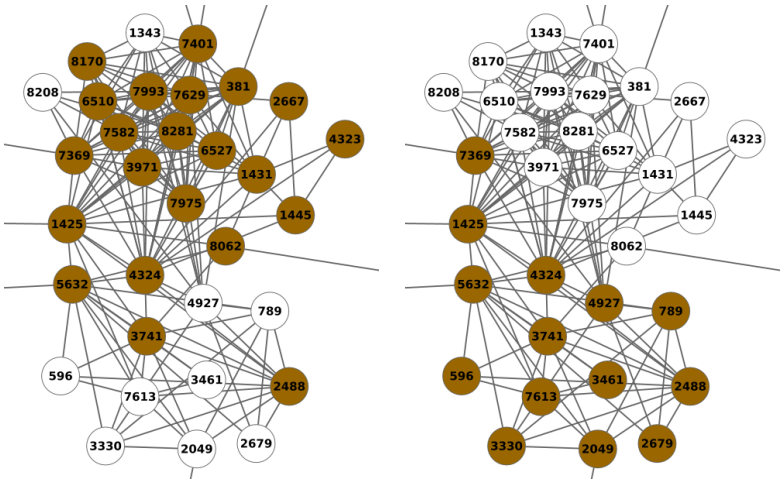
An example of this analysis is depicted in Figure 7.7. In Figures 7.7(a-e) we depicted the same set of 32 nodes with the edges connecting them in the Facebook dimension. Figures 7.7(a-b) depicts two communities extracted by Demon in the Facebook dimension by coloring in blue the nodes belonging to them, leaving in white the remaining nodes. We can see that the two communities share an overlap of six nodes.

In Figures 7.7(c-d) we depict two overlapping communities in the FourSquare dimension (highlighted with a brown color). Please note that the actual edges depicted are from the Facebook network, making the comparison of the nodes' community affiliations more clear. We can see that the two FourSquare communities are still overlapping, but with different common elements: by interacting with their Facebook friends that are part of the overlap in the Facebook direction, some users visit the same places of other users that are not directly their friend, creating a new community.

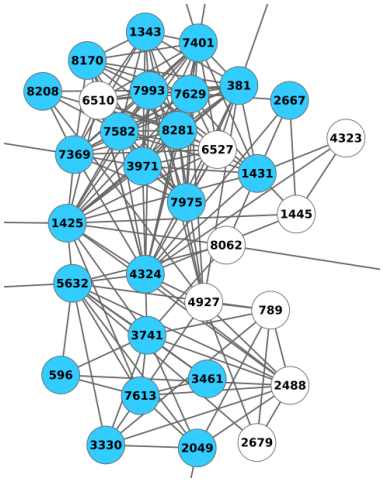
Finally, in Figure 7.7(e) we have the community extracted from Twitter network (light blue color). We can see that, in Twitter, the two overlapping communities are actually one single



(a) Facebook #1 (b) Facebook #2



(c) FourSquare #1 (d) FourSquare #2



(e) Twitter

Figure 7.7: The overlapping communities in the three dimensions of the network.

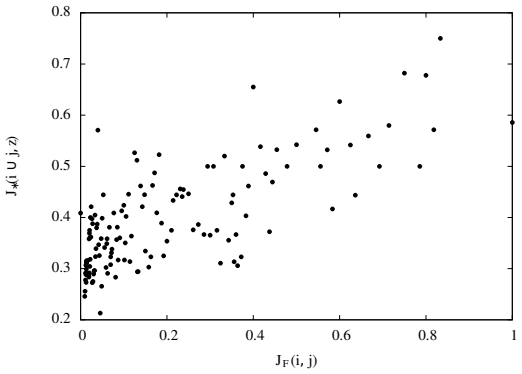


Figure 7.8: The relationship between $J_F(i,j)$ and $J_*(i \cup j, z)$ for the communities in the Facebook dimension.

community, with the exception of some nodes that do not use the Twitter service. The overlap in the Facebook dimension is likely to be playing a role here: even if user a is not friend of user b, he still may be interested in user b's tweets, as many of user a's friends are friend with user b.

This is only one example out of many that can be found. To prove this, we took each community couple (i,j) out of the 367 communities we found in the Facebook dimension. We calculated the Jaccard index $J_F(i,j)$ of the two communities, i.e., their degree of overlap. Then, we joined the two communities and we calculated the Jaccard index between the community union $i \cup j$ and each community in the Twitter and Foursquare dimension, i.e., how much the two overlapping communities are part of a single community in another dimension. We refer to this quantity as $J_*(i \cup j, z)$. We plot the relationship between $J_F(i,j)$ and $J_*(i \cup j, z)$ in Figure 7.8. Since many couples of communities may have the same $J_F(i,j)$ value, we took the average of all $J_*(i \cup j, z)$ for each $J_F(i,j)$ value. As we can see, the larger the overlap in the Facebook dimension, the more the two communities are included in a single community in another dimension.

The Demon-based Explanation of Communities

In the previous sections, we saw that the approach implemented by Demon is able to better uncover the community structure implied in real world networks. We can conclude that there is a correlation between how the overlap in communities forms in the real world and how Demon is able to detect the overlap. As a consequence, by explaining how Demon detects the overlap we may have an insight about how the overlap works in the real world.

We decided to translate this idea into a generative model. In other words, we can use the principle of Demon to generate synthetic networks. As a by-product, we will have benchmarks for other overlapping community discovery algorithms that reflect better the overlap mechanics of social networks, when these mechanics matches with our theory of connectivity driven by latent factors. This is a useful track of research as most of the benchmark networks for community discovery do not generate overlapping communities. The benchmark network presented in [182] does, and we used it in the previous section. However, there are some shortcomings included in that method. First, it is mandatory to specify in how many communities the overlapping nodes lie, and each node will belong to exactly the same number of communities, which is unrealistic. Second, it is mandatory to specify how many nodes are part of more than one community and how many are not, an information that is difficult to understand if we want to model real world networks.

Demon starts from the assumption that communities are generated locally around each node. Therefore, it expects to find in the ego network of each node a set of well-separated semi-cliques. We describe the Network Generator based on Demon in Algorithm 5. First, for each $v \in V$ we extract its number of neighbors, by generating a power law degree distribution with exponent α (step #1). So, for each node v we keep in $deg'(v)$ the number of connections that v is accepting.

Algorithm 5 The pseudo-code of the NeGen DEMON.

Require: α, β, γ, $|V|$
Ensure: set of nodes and edges $\mathcal{G} = (V, E)$
1: $V \leftarrow POWERLAW(\alpha)$
2: $SORTDEGDESC(V)$
3: **for all** $v \in V$ **do**
4: **if** $deg'(v) > 1$ **then**
5: $n \leftarrow deg'(v)$
6: $deg'(v) \leftarrow 0$
7: $N \leftarrow RANDOMSAMPLE(V, n)$
8: $E \leftarrow CONNECTNEIGHBORS(v, N)$
9: $E \leftarrow E \cup COMMUNITIES(v, N, \beta, \gamma)$
10: $UPDATE(deg'(N))$
11: **end if**
12: **end for**
13: **return** $\mathcal{G}$

Algorithm 6 The COMMUNITIES routine of NeGen DEMON.

Require: v, N, β, γ
Ensure: set of edges E'
1: $C \leftarrow NORMAL(v, \beta)$
2: **for all** $c \in C$ **do**
3: $m \leftarrow RANDOMSAMPLE(N, size(c))$
4: $E' \leftarrow RANDOMCONNECT(m, \gamma)$
5: **end for**
6: **return** E'

Then we cycle over the nodes, starting from the ones with higher degree (steps #2-3). We first check that the node v is still accepting connections (step #4). In steps #5-7 we randomly select from V $n = deg'(v)$ nodes that will be neighbors of v, and we set $deg'(v) = 0$. We connect these n nodes with v (step #8) and then we generate the communities around v using the function $COMMUNITIES$ (step #9). Of course this will modify the number of connections that can be still attached to the extracted nodes, thus we update their deg' values in step #10.

How we create the local communities in the ego network is the task of the $COMMUNITIES$ function, and it pseudocode is reported in Algorithm 6. First we define the distribution of the community size around a node. We assume this distribution to be normal with an average equal to β. Then, for each community we extract randomly a number of nodes equal to its size and we connect them with probability γ (that should be larger than 0.5). Using this algorithm, it is possible to generate well-separated local communities in the ego networks of each node. These communities will eventually overlap when each neighbor of a previously considered node will generate its own local communities.

Discussion

In this work we proposed to see the emergence of overlapping communities in complex networks as the effect of latent factors driving nodes' connectivity. Based on this assumption, we created a new method for solving the problem of detecting this latent knowledge from significant communities in complex networks. We propose a democratic approach, where the peer nodes judge if their neighbors should be clustered together. We extended previous work by creating a consistent theoretical ground for our method. Moreover, we extended the algorithm to find hierarchical communities and we have provided evidences that the underlying assumption of the work could be correct.

We have shown in the experimental section that this method allows a discovery of communities in different real world networks collected from information rich datasets. The quality of the over-

lapping coverage, a community organization that allows nodes to be in different communities at the same time, is improved w.r.t. state-of-the-art algorithms, evaluated using both a standard synthetic network generator and real world networks, in which we use the communities to predict the metadata attached to the nodes. We also show that the performances of the algorithm are useful to shed some light about how and why social communities are overlapping, by analyzing a multidimensional network and providing the intuition that Demon can give us about how overlapping communities form.

7.2 Homophily and Quantification[1]

In this work we tackle the problem of estimating the frequency of different classes in a node labeled network. To achieve such goal, we propose quantification techniques that exploit the homophily effect observed in many social networks [176, 191, 192, 144]: people tend mostly to relate with others whom they share some interests, ideas or beliefs. Starting from this observation, as a first step our approaches divide the original network in sub-networks in order to better bound homophily. In particular two different partitioning strategies will be analyzed: one exploiting community discovery while the other adopting the notion of ego-network. The former approach tries to estimate the class prevalence in a networked population taking into account the characteristics of communities composing the entire network and the class frequencies in each community. The latter conversely, partitions the network in ego-networks and tries to infer the class of each unlabeled node in the network by observing the class of its neighborhood.

Problem Formulation

Quantification is an important issue to tackle in order to understand and monitor user's behaviors and activities by using social media and web data (i.e., *Big Data*) as the source of information. Recently high-performing approaches based on decision tree variants[144] have been proposed in order to solve it in general contexts: in the following we formalize how it can be instantiated in the specific case of networked data.

We model the network as an indirect graph that we denote by $G = (V, E, L)$, where V is the set of labelled nodes, L is a set of classes, and E is a set of edges, i.e., a set of pairs (u, v) where $u, v \in V$ are nodes. The classes in L represent the potential node labels and a classifier f is a function $f : V \to L$ that assigns a class label $l_i \in L$ to each node $v_j \in V$.
The actual frequency of a class l_i with respect to a network $G = (V, E, L)$ is $freq_V(l_i) = \frac{|\{v_j \in V | v_j.class = l_i\}|}{|V|}$. The estimated frequency using the classifier f is $\widehat{freq}_V(l_i) = \frac{|\{v_j \in V | f(v_j) = l_i\}|}{|V|}$.
Given the set of nodes V, we denote by V_l the subset of labeled nodes while we use V_u to denote the set of unlabeled nodes. In the following we will sometimes use "test set" to indicate V_u. Note that in our setting the classifier is not trained in an offline phase, like in typical *eager* learners. Our function f defined above is an *instance-based* classifier, because it operates on the premises that classification of unknown instances can be done by relating the unknown to the known according to some specific relation between the two kind of instances. We use the standard notation to indicate the set of *true positives* (TP), *false positives* (FP), *true negatives* (TN) and *false negatives* (FN) of a binary classifier. We use $tpr = \frac{TP}{TP+FN}$ to denote the *true positive rate* and $fpr = \frac{FP}{TN+FP}$ to denote the *false positive rate*. Given these preliminaries, the problem of quantification on network data is defined as follows:

Definition 14 (Network Quantification Problem) *Let $L = \{l_1, l_2, \ldots, l_n\}$ be a set of classes. Given a network $G = (V, E, L)$ and a partition of the nodes V into labeled V_l and unlabeled V_u the network quantification problem consists in finding a classifier f for the best estimation of the class label distribution in V_u, i.e., $\forall l_i \in L$ we want to minimize the difference between the actual frequency $freq_{V_u}(l_i)$ and the estimated one $\widehat{freq}_{V_u}(l_i)$.*

The following example highlights the final goal of quantification by comparing it with the classification.

Example 1 *Consider the network in Figure 7.9(a) where colored nodes are those whose class values are known. Here, the real frequencies of the two classes are: $freq(A) = \frac{5}{11}$ and $freq(B) = \frac{6}{11}$. Suppose we now apply two different classifiers to predict the labels of the unlabeled nodes. Figure 7.9(b) and Figure 7.9(c) shows the results of the two classifiers, where the red dashed nodes*

[1] G. Rossetti, L. Milli, A. Monreale, D. Pedreschi, F. Giannotti and F. Sebastiani, "Quantification for Complex Networks via Homophily", 2014

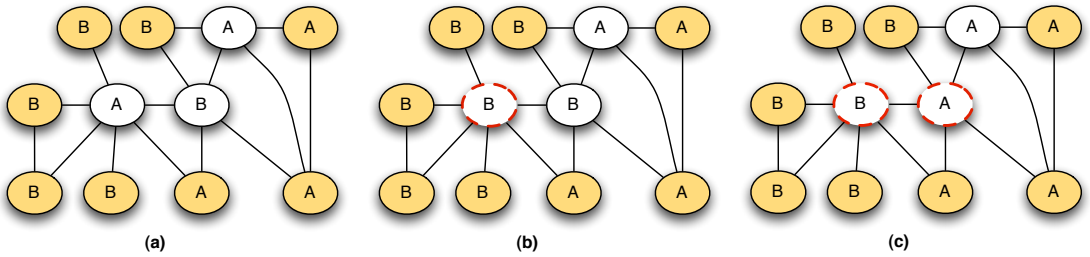


Figure 7.9: Network quantification vs Classification

represent the misclassified nodes. In Figure 7.9(b) the percentage of correctly classified node is $\frac{2}{3}$ *and* $\widehat{freq}(A) = \frac{4}{11}$*, so the total percentage of misclassified nodes is* $\frac{1}{11}$*. In Figure 7.9(c) we have two misclassified nodes with an accuracy of* $\frac{1}{3}$*, thus this result is worse than the previous one. However, if we focus on quantification, we have* $\widehat{freq}(A) = \frac{5}{11}$*, that is exactly the real frequency of class A, i.e., we do not have any quantification error.*

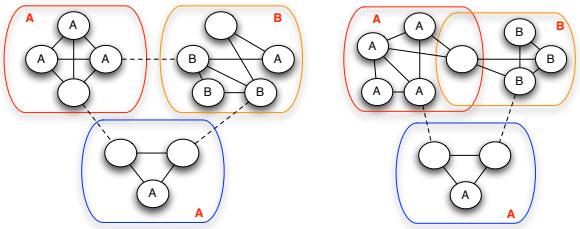
This example shows that in order to accurately quantifying the prevalence of classes we need to define ad-hoc techniques: even if there are some similarities with the classification task there is not equivalence between the quality of the respective results. A straightforward solution to the network quantification problem could be *sampling*, i.e., we could count the number of instances for each class value in the set of labeled nodes and assume that the same proportion of labels in the test set. However, this approach is not suitable when the class label distribution in the test set is different from the one observed in the training set, that is the case of real interest for quantification. This is the exact scenario depicted in our example. Indeed, we can see how sampling would return the result depicted in Figure 7.9(b). In detail, if we do not consider the unlabeled nodes, by sampling we obtain $freq(A) = \frac{3}{8}$ and $freq(B) = \frac{5}{8}$ while the real frequencies are $freq(A) = \frac{5}{11}$ and $freq(B) = \frac{6}{11}$.

Quantification methods via classification

The typical approach adopted in the literature, to address the quantification problem in non relational data, is based on standard *classification*. The idea in [135, 49, 50, 144] is to use a standard classifier and then post-process the results via specific methods to improve the quantification accuracy. Moreover, all the methods proposed so far solve that quantification via classification address only the binary class scenario: anyway, they can be easily extended to deal with single-label multi-class scenarios. More specifically, in [50] the following methods are introduced in order to post-process the results of classifiers and optimize them for quantification:

- **Classify & Count (CC).** This method once generated a classifier from the training set $\mathcal{T}_r$, and classified the unlabeled records in the test set, $\mathcal{T}_e$, estimates for each class l_i its frequency $freq_{\mathcal{T}_e}(l_i)$ by counting the fraction of records in $\mathcal{T}_e$ that have been labeled with c_i. We denote the computed estimation by $\widehat{freq}^{CC}_{\mathcal{T}_e}(l_i)$.
- **Adjusted Classify & Count (AC).** This methods attempt to improve the results obtained by the previous method by adjusting the quantification obtained by *Classify and Count* $\widehat{freq}^{CC}_{\mathcal{T}_e}(l_i)$ with the information about the true positive rate and false positive rate w.r.t. the training set: $\widehat{freq}^{AC}_{\mathcal{T}_e}(l_i) = \frac{freq^{CC}_{\mathcal{T}_e}(l_i) - fpr_{\mathcal{T}_r}}{tpr_{\mathcal{T}_r} - fpr_{\mathcal{T}_r}}$.

Both the methods can also be used in a network setting after the application of approaches tailored for network classification. In such scenario the training set $\mathcal{T}_r$ becomes V_l while the test set $\mathcal{T}_e$ becomes V_u. When a classifier provides a prediction score for each node in the network, classification-based quantification methods can be applied. However, standard classifiers, optimized for predicting the class of a single element, are not optimal for quantification.



(a) Neat partition (b) Smooth partition

Figure 7.10: Community based quantification without overlaps *(a)* and with overlaps *(b)*

Now we can introduce our quantification methods for networking data. These methods can be classified in two categories: approaches based on community discovery, and approaches based on ego-networks.

7.2.1 Community Discovery for Quantification

The methods in this category require the execution of two steps:

1. finding the set of communities;
2. assigning to each unlabeled node a class label by using the information extracted form the communities.

The first step of the algorithm is very simple. Given the whole network G containing the nodes of the training and test sets, we apply a community detection algorithm that finds clusters of nodes by taking into account the nodes' connections.

To perform the second step, for each community c_i the class label with the highest frequency is identified and assigned to each unlabeled node in the community. In detail, for each community c_i the algorithm computes the frequency of each class $freq_{c_i}(l_i)$, identifies the most frequent label (denoted by $Lmax_{c_i}$) and assigns it to each unlabeled node belonging to the community c_i.

To clarify how this approach works, in Figure 7.10 (a) we present a simple example where the algorithm of community discovery finds three communities (that we identify with the colors red, blue and orange). The second step our method, after having computed the frequency of each label within each community, will assign to the unlabeled nodes belonging to the red community the class label A, to the orange community the class label B and to the blue community the class A. Even if straightforward, this approach is not suitable when the community discovery algorithm returns overlapping communities as in the network depicted in Figure 7.10 (b). In these cases, a node can belong to several communities and each community may have a different majority class label. For example, in Figure 7.10 (b) we have a node belonging to the intersection between the red and orange communities. Moreover, the majority class in the red community is A while in the orange one is B. Now, the question is: *how can we decide the class label for the shared nodes?*

We propose two different strategies to decide which class label must be assigned to a node belonging to multiple communities:

- **Frequency**.
  The first strategy is to assign the class label with the greatest overall relative frequency in the labeled nodes. Therefore, if a node v_j belongs to m communities and the set of most frequent classes is $\{Lmax_{c_1}, Lmax_{c_2}, \ldots, Lmax_{c_h}\}$ $(h \leq m)$ then, the node v_j will have the label $Lmax_{c_i}$ if $freq_{c_i}(Lmax_{c_i}) = max_{c_i \in C}\{freq_{c_i}(Lmax_{c_i})\}$.

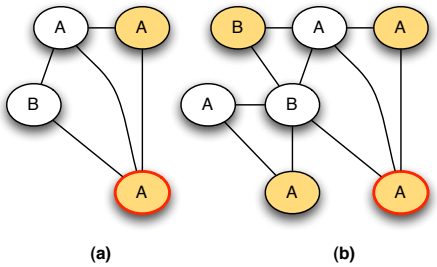


Figure 7.11: Ego-network at 1-hop and 2-hops

- **Density**.
  The second strategy is to assign the highest frequency class label of the denser community to which the node belongs.

In the above example, related to Figure 7.10(b), adopting the frequency strategy we get: in the red community $Lmax_{red} = A$ with frequency $\frac{4}{5} = 0.8$, while in the orange community $Lmax_{orange} = B$ with frequency $\frac{3}{4} = 0.75$. This implies that we will assign the label A to the shared node because it has a higher frequency. However, following the density policy we get: $density(red) = \frac{7}{10} = 0.7$, $density(orange) = \frac{5}{6} \simeq 0.83$, implying that the shared node will be assigned label B.

After the labeling we have two possibilities:

1. apply the *Classify and Count* strategy, i.e., we compute in the whole network the distribution of the class values by simple counting the nodes labelled with the same label;

2. apply the *Adjusted Classify & Count*, i.e., we adjust the quantification obtained by *Classify & Count* with the information about the true positive rate and false positive rate with respect to the nodes in the training set.

The weakness of the methodology described so far is that, if the network has some isolated and unlabeled nodes, we are not able to assign labels to them because they do not belong to any community. In these particular cases we decided to assign to those nodes a class label by following the same class distribution of the training set, i.e., if we have a known distribution of 0.4 for A and 0.6 for B in the training set the isolated nodes of the test set will be assigned respectively 40% to the former class and 60% to the latter.

7.2.2 Ego-networks for Quantification

An alternative set of methods that we propose in order to solve the problem of quantification on networks are based on the idea of assigning to each specific node the label that is the most frequent in its neighborhood. In this case we are directly exploiting the homophily property. This approach differs from the previous one, where the communities are found without considering the information about the node labels and using only the topological structure of the network. In the previous approach, only then, as a post-processing step a label assignment is performed using the homophily assumption within each community. The quantification process also in this case is composed of two main steps:

1. extraction of ego-networks for the unlabeled nodes, and

2. label assignment to the, unlabeled, central node.

The first step is to generate a partition of the network G into different subgraphs, called *ego-networks*. As seen in 7.1, an ego-network is a sub-network centered on a particular node who is the subject of the network. The focal point of the network is called the *ego*. In an ego-network, only nodes that are directly connected to the ego form the extracted substructure. An ego-network

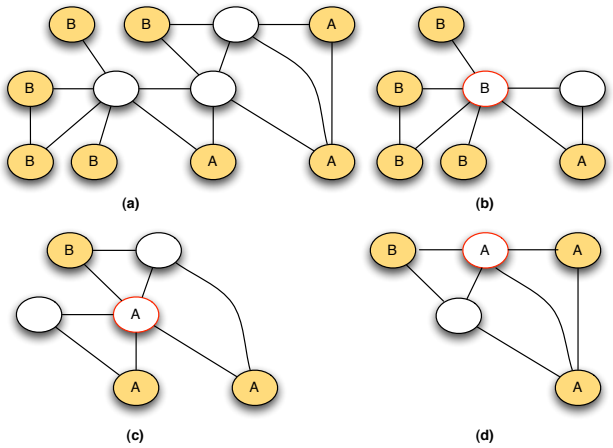


Figure 7.12: Ego-network based quantification. *(a)* the full graph with labeled and unlabeled nodes; *(b-c-d)* 1-hop ego-networks of the unlabeled nodes.

enables a focused view on the specific properties of a node, highlighting all its interactions with the neighbors. Figure 7.11 depicts an ego-network example, showing the relations of a node (the ego) and its neighbors. Obviously, the definition of ego-network can be extended by taking into account the k-hop neighborhood; in other words, the size of the ego's neighborhood is expanded by including all nodes to whom the ego has a connection at a path length of k, and all the connections between all of these nodes. Intuitively, the more k grows the more the homophily tends to decrease. In our approach, for each node of the test set we extract its ego-network at k-hops.

After this step, for each ego-network $EG = (V', E', L)$ the algorithm computes the frequency of each class $freq_{V'}(l_i)$ and then identifies the most frequent one denoted by $Lmax_{EG}$. Finally, $Lmax_{EG}$ is assigned to the ego node. Also in this case we can have some isolated and unlabeled node that make hard the assignment of the class label because it has no neighbors. As in the previous method, in these particular cases our strategy is to assign to those nodes the class label by following the same class distribution of the training set. After the assignment of the label to the node, we can apply the *Classify & Count* strategy or the *Adjusted Classify & Count* strategy.

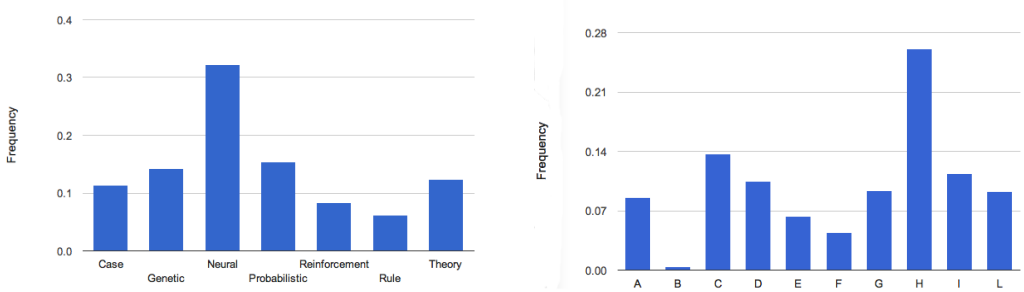
To clarify how this approach works, we discuss a simple example depicted in Figure 7.12. This figure illustrates the whole process of the algorithm considering ego-networks at 1-hop. In particular, Figure 7.12(a) depicts the original network where the white nodes have unknown class label. This network is the same used in Figure 7.9. The first step is to extract the ego-networks at 1-hop for each unlabeled (white) node. Figures 7.12 (b),(c) & (d) show the result of this step. Note that the ego node is indicated by a red border. Then, to each ego node the algorithm assigns the computed label. As a consequence, to the ego node in Figure 7.12(b) we assign the label B, while to the ego nodes in Figure 7.12(c) and Figure 7.12(d) we assign the class label A. In this case, this allows us to obtain a perfect quantifier even if the classification of each node is not perfect, as highlighted in Example 1.

Experiments

Here we present the evaluation of our methods and the results obtained from our experimentation. Firstly, we provide a description of the data and community discovery algorithms used in our experiments, then we show and discuss the obtained results.

Datasets

In the evaluation of our methods we used the following datasets:



(a) CoRA (b) Google+

Figure 7.13: Label Frequencies. *(a)* CoRA, *(b)* Google+

- *CoRA*: This dataset comprises computer science research papers: the network is built over papers using as relations for the edges both citation and shared-authors. The number of possible different class labels are 7 (their distribution is reported in Figure7.13)(a). The network contains 4,240 nodes and 77,824 edges.
- *IMDb*: This dataset is extracted form the Internet Movie Database, and contains description of movies released in the United States between 1996 and 2001. The class identifies whether the opening weekend box-office sales, have exceeded $2 million (class distribution: 57% and 43% respectively). In our network movies are linked if they share a production company, producer, director, or actor. The network contains 1,440 nodes and 51,481 edges.
- *Google+*: Social network built on the Google+ service extended with semantic information. The class labels identify the schools attended by the analyzed users (label distribution is reported in Figure7.13(b). Different schools are identified with letters from A to L.). Our network contains 33,381 nodes and 110,142 edges [193].

Community Discovery Methods.

To evaluate our quantification methods based on community discovery in our experiments we use two different algorithms for the detection of communities: Demon (introduced in 7.1) and Infohiermap [194]. They provide the opportunity of testing both the case of overlapping communities (Demon) and the case of non-overlapping ones (Infohiermap). We use Demon to extract both the communities and the micro-communities (the outcome of the local extraction without the application of any merging procedure). In following only the results of the micro-communities ones is discussed: this choice was made to reduce the average community size while amplifying the homophily effect.

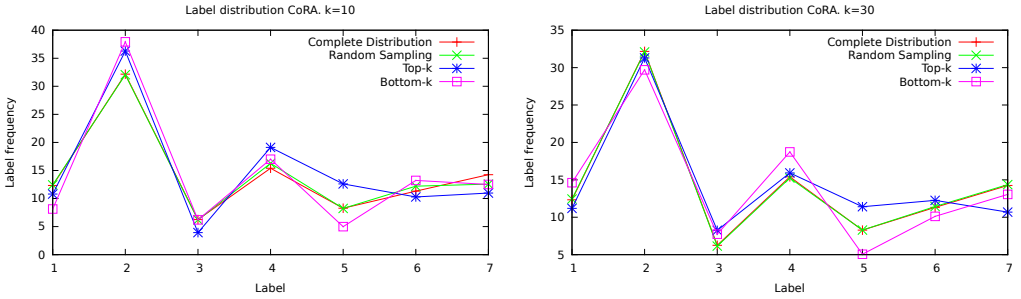
Empirical Evaluation

Our evaluation is organized as follows. First, the compared quantifiers are introduced, and highlights on their implementation are given. Then, the measure used to evaluate their accuracy is explained; three different strategies for building the test set are proposed and the experimental results are presented. Moreover, we discuss how network *assortativity* influences the results of the proposed quantification methods. Lastly, we analyze the effects of the overlaps on the performances obtained by community-based approaches.

Network Quantifiers

We compare, on the previously introduced networks, seven algorithms:

- Community Discovery based quantification by Infohiermap and Demon;



(a) (b)

Figure 7.14: Label frequency distribution on CoRA for *(a)* $k = 10\%$ and *(b)* $k = 30\%$.

- *EG*: Ego-network based labeling (Ego-networks were extracted at one and two hops);
- *LBQ*: Link-based quantification, as defined in [51];
- *wvRN*: network classifier, as defined in [195]; and,
- *Baseline*: sampling.

The methodologies we propose were tested in their two variants: *Classify & Count* and *Adjusted Classify & Count*. In order to compute the latter for each node $n \in V_l$ the proposed algorithms were applied to newly identify its label: in this way, following a *leave-one-out* strategy, is possible to compute the True Positive Rate and the False Positive Rate on V_l: estimates of tpr and fpr are needed to adjust the label frequency obtained by the standard *Classify & Count*. Once computed the new frequencies, a rescaling step is applied to assure that, for each network, the sum of labels' frequencies is equal to one.
It is worth to noting that the Link-based quantification approaches (LBQ), discussed in the related work section 4.2.2, was slightly modified: the frequencies of labels were assigned using not the median but the mean value of the distribution and, as done for the *Adjusted Classify & Count*, at the end a normalization step was applied to overcome the highlighted issue (i.e. the sum of the estimated frequencies for labels needs to be equal to one).

Kullback-Leibler Divergence.

In order to evaluate the accuracy of a quantifier we need to compare $\widehat{freq_V}(l_i)$, the frequency computed for the new labeled nodes l_i, with $freq_V(l_i)$, its actual frequency. Different measures have been used in the literature for measuring quantification accuracy: the most convincing among the ones proposed so far is the one used by Forman in [50], which uses normalized cross-entropy, better known as Kullback-Leibler Divergence (KLD), defined as:

$$KLD\left(freq_{\mathcal{V}}(l_i)||\widehat{freq_{\mathcal{L}}}(l_i)\right) = \sum_{i=1}^{n} freq_{\mathcal{V}}(l_i) \log \frac{freq_{\mathcal{V}}(l_i)}{\widehat{freq_{\mathcal{V}}}(l_i)} \tag{7.11}$$

KLD aims at evaluating the information loss when $\widehat{freq_V}(l_i)$ is used as approximation of $freq_V(l_i)$. It ranges in the interval $[0, +\infty)$: 0 means that the two frequency values are equal for each l_i and $+\infty$ means that their values diverge. If $\widehat{freq_V}(l_i) = 0$ for at least one class, KLD is not defined: therefore, as in [50], we add a small amount ϵ (set to $\frac{0.5}{|V_u|}$) to both numerator and denominator in the *log* function.

	CoRA			CoRA Bottom			CoRA Top		
Method	10	20	30	10	20	30	10	20	30
INFOHIERMAP	1.369e-2	3.276e-2	4.026e-2	4.213e-2	2.895e-2	3.737e-2	1.211e-2	1.890e-2	1.671e-2
DEMON	1.777e-2	2.578e-2	3.830e-2	4.779e-2	3.671e-2	4.173e-2	1.008e-1	5.255e-2	6.055e-2
DEMON Density	5.425e-3	1.7375e-2	3.273e-2	5.238e-2	3.695e-2	4.627e-2	5.528e-2	6.258e-2	7.012e-2
EG1h	6.719e-3	1.533e-2	3.003e-2	5.194e-2	3.951e-2	4.762e-2	5.124e-2	6.298e-2	5.958e-2
EG2h	1.459e-2	1.149e-2	4.081e-2	**1.909e-2**	2.111e-2	2.486e-2	1.408e-1	1.184e-1	1.310e-1
INFOHIERMAP AC	6.2387e-3	1.324e-2	1.902e-2	3.724e-2	2.401e-2	3.454e-2	**7.219e-3**	1.593e-2	**1.453e-2**
DEMON AC	1.031e-2	1.118e-2	1.648e-2	4.290e-2	3.277e-2	3.890e-2	9.589e-2	4.968e-2	5.841e-2
DEMON Density AC	2.276e+0	1.290e-1	1.307e-1	1.170e-1	2.574e+0	1.199e+0	2.305e+0	2.173e+0	1.191e+0
EG1h AC	**1.197e-3**	**4.395e-3**	**3.207e-3**	2.884e-2	2.386e-2	3.926e-2	3.152e-2	**1.451e-2**	1.730e-2
EG2h AC	3.354e-3	5.484e-3	6.124e-3	2.926e-2	**2.045e-2**	**2.587e-2**	1.195e-2	1.522e-2	2.246e-2
LBQ1h	1.990e-1	2.523e-1	2.333e-1	3.132e-1	2.750e-1	2.763e-1	3.634e-1	3.251e-1	2.733e-1
LBQ2h	1.927e-1	2.478e-1	2.313e-1	3.132e-1	2.772e-1	2.531e-1	2.969e-1	3.273e-1	2.972e-1
Baseline	7.450e-3	1.512e-2	3.126e-2	5.285e-2	4.789e-2	5.890e-2	6.427e-2	7.447e-2	7.789e-2
wvRN	2.773e-2	1.684e-2	2.349e-2	1.353e-1	6.965e-2	4.214e-2	1.159e+0	7.229e-2	7.475e-2

Table 7.8: CoRA: mean of KLD of predicted and actual quantification for all quantifiers.

	IMDb			IMDb Bottom			IMDb Top		
Method	10	20	30	10	20	30	10	20	30
INFOHIERMAP	7.948e-3	1.918e-2	2.124e-2	1.075e-1	1.384e-1	2.013e-1	8.496e-3	6.752e-3	3.047e-3
DEMON	1.321e-1	1.599e-1	1.759e-1	2.570e-1	4.254e-1	5.762e-1	5.177e-3	1.617e-2	6.159e-2
DEMON Density	5.451e-2	6.009e-2	2.171e-2	2.777e-1	4.067e-1	4.882e-1	4.615e-1	5.638e-1	4.365e-1
EG1h	1.716e-1	1.916e-1	1.065e-1	2.856e-1	4.215e-1	5.029e-1	5.155e-1	5.761e-1	5.162e-1
EG2h	7.248e-1	7.684e-1	5.248e-1	1.681e+0	2.482e+0	2.694e+0	1.638e+0	1.463e+0	1.407e+0
INFOHIERMAP AC	7.830e-4	4.996e-3	2.188e-4	1.024e-1	1.353e-1	1.991e-1	3.328e-3	**3.661e-3**	**8.415e-4**
DEMON AC	1.248e-1	1.457e-1	1.549e-1	2.519e-1	4.224e-1	5.740e-1	**8.824e-6**	1.308e-2	5.938e-2
DEMON Density AC	1.267e-2	1.202e-2	1.108e-2	**2.908e-3**	**3.852e-4**	**7.043e-3**	2.040e-2	4.241e-2	6.925e-2
EG1h AC	**4.525e-4**	1.490e-4	**5.560e-5**	7.915e-1	6.056e-1	3.593e+0	2.060e-2	1.211e-2	4.608e+0
EG2h AC	1.359e+0	5.452e-1	3.711e+0	7.915e-1	6.056e-1	3.593e+0	2.465e+0	1.915e+0	1.795e+0
LBQ1h	6.883e-4	**7.872e-5**	5.674e-3	2.314e-1	1.955e-1	1.793e-1	1.076e-1	1.176e-1	1.010e-1
LBQ2h	1.052e-2	3.967e-3	8.692e-3	1.323e-1	7.828e-2	3.484e-2	2.069e+0	6.573e-2	6.638e-2
Baseline	8.461e-3	1.427e-2	2.154e-3	2.750e-1	4.360e-1	4.872e-1	8.468e-2	2.104e-1	2.568e-1
wvRN	3.196e-3	6.541e-3	7.053e-4	4.486e-1	5.672e-1	5.368e-1	1.432e-1	2.257e-1	3.309e-1

Table 7.9: IMDb: mean of KLD of predicted and actual quantification for all quantifiers.

Test Set Scenarios.

To better characterize the performances of the compared methodologies we have identified three different scenarios:

- *Random*: the $k\%$ unlabeled nodes are chosen uniformly at random from the whole network;
- *Top*: the chosen nodes are the *top-k*% w.r.t. the degree distribution;
- *Bottom*: the chosen nodes are the *bottom-k*% w.r.t. the degree distribution.

Our aim is to capture the average scenario (*Random*) and two more complex ones which identify those cases in which the neighborhood of unlabeled nodes offers too little (*Bottom-k*) or too much (*Top-k*) information to be exploited for assigning labels. Furthermore, we test all the algorithms by fixing, for each network, the ratio of unlabeled nodes to 10%, 20% and 30% of $|V|$. As shown in Figure 7.14, for the CoRA network (as well as for all the other datasets analyzed) the label frequency distributions computed on the *Top-k* and *Bottom-k* node samples show variation w.r.t. the one computed on the whole dataset. Conversely, for the *Random* sampling the frequency distribution does not differ significantly from the complete network's one. We report for each network a table with the KLDs score of the tested approaches: the best results are highlighted in bold for each value k and node sampling scenario.

- *Random sampling*:
  Extracting $k\%$ of the nodes uniformly at random from the original network ensures that the label distribution of V_u shows a very low drift from the one of V_l. This scenario is very unlikely on real data and in this case also simple approaches, as sampling, can produce good

	Google+ School			Google+ School Bottom			Google+ School Top		
Method	10	20	30	10	20	30	10	20	30
INFOHIERMAP	5.723e-3	6.247e-3	6.126e-3	7.281e-3	5.620e-3	3.475e-3	4.862e-4	**5.511e-4**	2.011e-3
DEMON	8.370e-3	1.309e-2	1.517e-2	2.375e-2	3.563e-2	4.276e-2	1.936e-3	1.638e-3	**3.452e-4**
DEMON Density	8.710e-3	1.393e-2	1.661e-2	2.375e-2	3.563e-2	4.251e-2	2.159e-3	1.350e-3	3.875e-3
EG1h	1.366e-3	1.685e-3	1.823e-3	6.584e-3	5.154e-3	6.316e-3	8.580e-4	5.801e-3	1.719e-2
EG2h	2.255e-3	5.116e-3	5.017e-3	9.139e-3	7.984e-3	6.816e-3	**3.752e-4**	1.042e-3	1.159e-3
INFOHIERMAP AC	8.770e-3	2.744e-3	2.641e-3	6.004e-1	5.385e-1	4.822e-1	9.563e-1	1.061e+0	9.135e-2
DEMON AC	3.047e-1	3.092e-1	3.150e-1	2.912e-1	2.840e-1	2.816e-1	2.910e-1	2.152e-1	1.887e-1
DEMON Density AC	3.038e-1	3.081e-1	3.138e-1	2.900e-1	2.830e-1	2.810e-1	2.927e-1	2.202e-1	1.989e-1
EG1h AC	**4.333e-4**	**1.160e-3**	**1.354e-3**	4.573e-3	3.424e-3	2.508e-3	1.411e-2	1.860e-3	2.531e-3
EG2h AC	7.465e-4	3.504e-3	4.177e-3	**2.913e-3**	**2.531e-3**	**1.795e-3**	3.110e-2	1.407e-2	1.167e-2
LBQ1h	2.867e-1	2.590e-1	2.623e-1	2.622e-1	2.512e-1	2.472e-1	4.284e-1	3.555e-1	3.163e-1
LBQ2h	2.887e-1	2.581e-1	2.612e-1	2.543e-1	2.416e-1	2.376e-1	4.273e-1	3.555e-1	3.146e-1
LBQ3H	2.830e-1	2.538e-1	2.555e-1	2.518e-1	2.390e-1	2.360e-1	4.221e-1	3.522e-1	3.138e-1
Baseline	1.772e-3	3.396e-3	5.385e-3	2.375e-2	3.563e-2	4.261e-2	1.753e-1	1.180e-1	8.617e-2
wvRN	2.206e-3	7.085e-3	4.391e-3	1.294e-2	1.657e-2	1.338e-2	1.881e-1	1.069e+0	4.037e-1

Table 7.10: Google+: mean of KLD of predicted and actual quantification for all quantifiers.

results. In CoRA, as well as in IMDb and Google+ (Table 7.8, 7.9 and 7.10), we observe how EG approaches outperform both the baseline and the community-based methods.

- *Bottom-k sampling*:
  Populating V_u with the *Bottom-k* nodes w.r.t. the degree distribution of the network may introduce a drift on the label frequency: this is reflected by the results provided by the baseline and LBQ which, increasing the size of the sample worsen their KLD. Conversely, our approaches tend to increase their performances as the size of V_u increases: this is due to the greater connectivity that can be exploited to assign labels. In CoRA and Google+ is again EG that registers the best KLD values, while on IMDb a community-based method (DEMON) is able to obtain the best performances.

- *Top-k sampling*:
  Similarly to the *Bottom-k*, the *Top-k* node sampling introduces a distribution drift due to the unequal probability for each node to be part of the V_u set. Contrary to the previous sampling strategy, the real challenge here is to correctly discriminate the information given by the high connectivity of the unlabeled nodes. The selected nodes are hubs: their high degree increases the probability of being connected with nodes that do not share common labels. We can observe how Baseline as well as LBQ and wvRN are not able to record the best performances: community-based approaches on CoRA, IMDb and Google+ show the overall better accuracy.

Homophily estimation: Label Assortativity

In order to justify the results given we analyzed the degree of homophily of our three datasets using a network measure called *assortativity*. Assortativity, or assortative mixing, measures the preference of a node to attach to others that are similar in some way, for this reason it can used as a proxy to estimate the overall homophily level within a network w.r.t. a specific feature (i.e., node degree). Due to the problem addressed and to the three proposed test set construction strategies we will focus on a specific instantiation of this measure: Label Assortativity which measures how much nodes tend to being connected with similar labeled ones. To compute assortativity is commonly used the Pearson correlation coefficient, that lies in $[-1, 1]$: a positive value indicates a correlation between nodes with similar labels, while a negative one denotes relationships between nodes with different labels. When the correlation is equal to 1, the network has a perfect assortative mixing patterns, when it is equal to 0 the network is non-assortative, while when it is equal to -1 the network is completely disassortative. CoRA and Google+ have high Label Assortativity (0.6233 and 0.8912, respectively): this reflects positively on the results of the EG quantifiers, which (almost always) outperform the other approaches, exploiting directly the homophily property. Instead, on IMDb which has a lower Label Assortativity (0.2787), i.e., lower homophily, community-based

approaches, which make use of broader information for assigning class labels, outperform the other quantifiers.

Overlapping Community Discovery.

Comparing the proposed community-based algorithms on the datasets with lower values of Label assortativity, we can notice a significant predominance of INFOHIERMAP KLD scores over the DEMON's ones. Given the high KLD values of the latter we may conjecture that overlaps can be a cause of misclassification: this result is supported by the fact that avoiding the merging phase to reduce the community's size (as well as their overlaps) we are able to improve DEMON performances on all the datasets.

Discussion

In this work we have proposed two approaches for performing quantification in complex networks. Our quantification methods exploit the homophily effect observed in many social networks. The first method, based on community discovery, estimates class prevalence in a population by considering the characteristics of the communities composing the entire network, while the second approach infers the class of each node in the network by observing the class distribution in its neighborhood. The thorough experimental evaluation that we have carried out shows that our methods outperform state-of the-art quantifiers.

Given the ever-growing availability of social networks and social media, solving the quantification problem on networks opens up new ways for the estimation of social indicators based on Big Data, provided that we can rely on relatively small surveys of labelled data. Moreover, in this work we have observed how the proposed approaches are stable to variations on the criteria used to build the test set: this is very important because quantification plays its major role when observing dynamic networks. In such scenarios, make assumption on the class distribution of novel nodes is not an easy task: being able to continuously providing such a valid estimate can be crucial to support decision processes.

7.3 Social Engagement: Skype[1]

As the social media space grows more and more people interact and share experiences through a plethora of different online services, producing every day a huge amount of personal data. Companies providing social media services are interested in exploiting these Big Data to understand the "user engagement", i.e., the way individuals use the products they provide. Predictive analytics, for example, allow for looking at historical patterns to make predictions about the future product usage of individuals, or for targeting the most influential customers for online advertisement and marketing purposes.

Traditional approaches of predictive analytics focus on *individuals*: they try to describe and predict the level of engagement of a single individual, with the purpose of suggesting proper products/services and favoring the diffusion of the system over a larger population. Focusing on individuals, however, introduces many challenging issues. First, the amount of individuals to process is enormous, and hence hardly manageable. Think about online giants like Skype or Facebook whose products where used regularly by millions of users every day. In these contexts, being able to provide an up-to-date description and prediction of user engagement for all the users is not practically feasible. Addressing each single individual is also in many cases redundant, since neighbors in networks tend to behave in a similar way showing a certain degree of homophily (as previously discussed in 7.2). A second issue about the user-centric approach is that it reaches poor performances in the classification and prediction of the individuals engagement. Restricting the analysis to single users inevitably causes the underestimation of the surrounding social context, whereas online social services are usually designed to foster social interactions between groups of individuals. It is hence fundamental to widen the analysis spectrum in order to incorporate social surrounding of users and, doing so, to capture the homophily which characterize real social networks.

We propose to shift the focus from individuals to groups, i.e., to analyze and describe the engagement of *social communities*. If user-centric approaches fail because they do not take into account the individuals' social surroundings, on the other hand, it goes without saying that analyzing the user engagement problem on the overall network does not make sense. The group-centric approach focuses on social communities as a trade-off between the micro and the macro level of network granularity (Figure 7.15). Moving the interest from individuals to communities brings many advantages. First, we reduce by several orders of magnitude the space of analysis, shrinking the number of objects to process and speeding up the analytical tasks. Second, targeting communities allows for capturing the homophily inherent to the social network: we can "compress" into one object all the densely connected components of a social group. Finally, groups are complex objects from which we can extract a wide set of features for the analysis.

In this work, we want to show that the structural and semantic informations attached to communities extracted from the Skype social network are able to characterize group engagement. Here, we do not address the problem of forecasting user engagement, as done in literature, but the issue of *describing* it: given the features of a social community we aim to classify the level of activity of that community. In order to compute social groups we choose four different approaches:

- two community discovery algorithms: one which maximize the partition density (HDemon (the hierarchical version of Demon discussed in 7.1) and the other its modularity (Louvain [110]);
- two baseline methods which, in turn, define communities as (i) the ego-network of individuals; (ii) composed by a fixed number of nodes discovered through a BFS visit.

Starting from the obtained community sets we computed an ensemble of features describing structural and geographical characteristics for each community. We then learned on them a classifier

[1] G. Rossetti, L. Pappalardo, R. Kikas, D. Pedreschi, F. Giannotti and M. Dumas, "Community-centric analysis of user engagement in Skype social networks, 2015

that reaches good performances when used to describe the engagement of social groups for the Skype Video and Chat services.

We find that group-centric approaches outperforms user-centric ones when we use algorithms producing overlapping micro-communities, like the DEMON algorithm. In contrast, adopting partitioning algorithms which maximize modularity and produce macro-communities, like LOUVAIN, the reached performances are worse than the ones of classical user-centric strategies. Hence, we show how the choice of a proper community detection algorithm is crucial to reach high performances in the engagement prediction. Moreover, we highlight the role played by the communities structural and geographical features in the predictive task by analyzing two different scenarios: when the target classes representing the level of user engagement are balanced and when they are not.

7.3.1 Understanding Group Engagement

Characterizing user engagement from social network information is a complex task. In many applicative scenarios, individual-based approaches for user engagement are not able to produce good accuracy: the main reason for their performances has to be identified in the lack of consideration given to the social tissue that surrounds each user. On the other hand, approaching the service engagement problem globally studying the network as a whole often leads to the impossibility to scale down the obtained results on well-defined substructures.

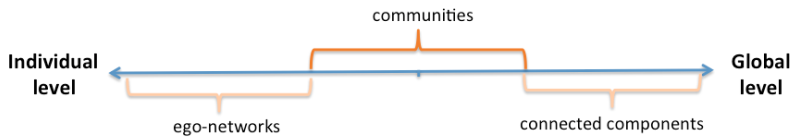


Figure 7.15: Interpolation between the local and the global level through different sizes network partitions.

Starting from such observations, we guess that a better descriptive power can be achieved when a compromise between the *micro* and the *macro* level of network granularity is reached. As shown in Figure 7.15, there are several ways to interpolate between the individual and the global level of a network: starting from the analysis of ego-networks, which capture the immediate surroundings of a single node, passing through social communities of different sizes and topological characteristics, till reaching the analysis of the connected components which constitute network natural partitions. Identifying the most adequate granularity means capturing social groups where nodes show the best similarity. Homophily in social contexts, indeed, acts as a strong glue: as we have already shown in 7.2, a partition which maximizes the homophily among members of the same group is able capture meaningful information about both individuals and their social surroundings.

Skype Dataset

We analyze a dataset of users and connections in the Skype network as of October 2011. The dataset includes anonymized data of the Skype users. Each user (identified by hashed ID) is associated with his account creation date, country and city.

The dataset also includes connections between users. Connections are undirected: a link exists between two users if and only if they belong to each other's contact list. Connections are established as follows: If a user u wants to add another user v to his contact list, u sends v a contact request. The connection is established at the moment v approves the request (or not established if the contact request is not approved). In the dataset, each connection is labeled with a timestamp corresponding to the contact request approval.

In addition to non-identifiable user profile data and network data, the dataset includes data about usage of two VOIP products: video calling and chatting. Product usage is aggregated

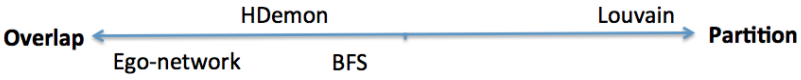


Figure 7.16: Characterization of the analyzed approach.

monthly. Specifically, for each product, for each user and for each month, we are given the number of days in the month when said user used the product in question. The product usage data does not provide information about individual interactions between users, such as participants in an interaction, content, length, or time of the interaction. The frequency of product usage is not recorded at a finer granularity than monthly.

In this work, we focus on analyzing the most recent available snapshot of the network. Accordingly, we focus on the subset of the dataset containing only users who used one of the two products, during at least two of the last three months covered in the dataset. Our analyses will be then executed on a filtered dataset composed by several tens of millions of users and connections.

Community Discovery

In our analysis we have considered different ways to group users of a social network, and thus we have made use of several algorithms each one defining a different kind of communities. As stated before, community discovery aims to identify tightly connected sets of users. The extensive literature on this subject have shown how, slightly varying the community definition, it is possible to produce different models that exhibit peculiar traits. In particular, the degree of *overlap* is a property that discriminates between CD algorithms. Classical approaches produce a partition of the network, i.e., an individual can be involved in at most one community. Overlapping approaches considers instead the multidimensional nature of social networks allowing the individuals to belong to many different communities.

We use four different algorithms to extract social communities from the Skype network: Louvain, HDemon, Ego-network and BFS. Such algorithms cover several declinations of overlap. Figure 7.16 orders the algorithms according to these property. HDemon, Ego-network and BFS cover several degree of overlap, while Louvain is the prototype of a partitioning modularity-driven algorithm. In Particular:

The Louvain algorithm [110], which is based on a greedy modularity approach, is fast and scalable on large networks and reach high accuracy on ad hoc modular networks. The modularity optimization is performed in two steps. First, the method looks for "small" communities by optimizing modularity locally. Second, it aggregates nodes belonging to the same community and builds a new network whose nodes are communities. These steps are repeated iteratively until a maximum of modularity is obtained, producing a hierarchy of communities. Louvain produces a complete non-overlapping partitioning of the graph. It has been shown that modularity-based approaches suffer a resolution limit and, therefore, Louvain is unable to detect medium size communities [109]. This will reflect in an average high density of its community due to the identification of a predominant set of very small ones (usually composed by 2-3 nodes) and a few very big ones. The algorithm is parameter-free: on the analyzed data it has produced a community hierarchy of seven levels.

HDemon, described in 7.1, is the hierarchical extension of the Demon algorithm: it is based on recursive aggregation of denser areas extracted from ego-networks. Its definition allows to compute communities with high internal density and tunable overlap. In its basic first hierarchical level, HDemon operates extracting ego-networks and partitioning them into denser areas using Label Propagation. The communities computed at a given hierarchical level are subsequently used as meta-nodes to build a new network in the next hierarchical level, in which the edges between meta-nodes are weighted using the Jaccard of meta-nodes' contents. This procedure stops when

disconnected meta-nodes, identifying the components of the original network, are obtained. The algorithm has two parameters, which are the minimum community size μ; and the minimum Jaccard ψ among meta-nodes to create an edge that connects them. We applied HDemon on the data fixing $\mu = 3$ (we consider a triangle the minimum community) and using two different values of the ψ parameter: $\psi = 0.25$ which produced the HDemon25 community set, and $\psi = 0.5$ which produced the HDemon50 community set. For each community set we consider only the first 5 levels of the produced community hierarchy.

Ego-network is a naive CD algorithm which models the communities as the set of induced subgraphs obtained considering each node with its neighbors. This approach represents the first step of the HDemon algorithm and provides the highest overlap among the four considered approaches: each node u belongs exactly to $|\Gamma(u)| + 1$ communities, where $\Gamma(u)$ identify its neighbor set. To avoid this extreme scenario, in the following we apply a node sampling strategy and consider only a ratio ϵ of the ego-networks for the analysis. We set the parameter $\epsilon = 0.2$, and randomly extracted a number of users equals to the 20% of the population. For each random user we extracted the corresponding ego network, filtering only unique ones (two users can have equal ego networks if they share all their contacts).

The BFS approach extracts random connected components from the graph. It randomly samples a ratio ϵ of the nodes of the network and, for each one of them, a number *csize* is extracted from a distribution, representing the distribution of community size. A breadth first search which explores *csize* nodes starts from a root: all the nodes obtained during each single search from a community. If the search produces a community smaller than *csize*, the community is discarded. This approach produces overlapping communities whose density tend to be low for high values of *czise*. BFS takes in input three parameters: the sample size ϵ, the exponent β and the cutoff τ of a power law distribution, representing the community size distribution (which is known to be a power law for many known community detection algorithms). We set $\epsilon = 0.2$ (20% of the population) and set the power law parameters to $\beta = 1.8$ and $\tau = 10,000$.

algorithm	**parameters**	**#sets**
Louvain	no parameters	7
HDemon	$\mu = 3$, $\psi = 0.25, 0.5$	10
Ego-Networks	$x = 0.2$	1
BFS	$\epsilon = 0.2$, $\beta = 1.8$, $\tau = 10,000$	1

Table 7.11: Community sets produced by the four CD algorithms.

Each algorithm produces different community sets when applied on the Skype dataset with the specified parameters (see Table 7.11 and Table 7.12). More specifically, in Table 7.12 are reported for each community set and hierarchy level ($Lv.$): (i) the number of communities ($\#C$); (ii) the induced node coverage w.r.t. the whole graph; (iii) the average number of communities per node (σ, i.e., the mean degree of overlap); the average community size ($Avg.size$). Louvain is a partitioning algorithm and guarantees the complete coverage of the nodes. HDemon covers around 76% of the nodes because imposing the parameter $\mu = 3$ we exclude communities with two nodes only. BFS and Ego-Network are executed on a 20% sample of the nodes, on which they cover the 90% and 69% respectively. For the Louvain community sets, we consider the hierarchical levels 0 and 6 only, which correspond to the first greedy iteration and the iteration having the maximum modularity. Figure 7.17 shows the community size distributions produced by the CD algorithms described above. They follow a power-law trend indicating that, regardless the community detection algorithm, a big heterogeneity characterize the size of social groups: most of the communities contain a few nodes, while a small but significant fraction of communities contain

[2]For Ego-Nets and BFS the coverage is computed starting from a 20% sample of the total users.

COMMUNITY STATISTICS

Algorithm	**Lv.**	$\#C$	**coverage (%)**	σ	Avg. size
HDEMON25	0	1.1e+08	76	13.1	8.2
	1	6.2e+07	76	23.2	26.0
	2	3.3e+07	76	13.2	27.9
	3	2.7e+07	76	6.3	16.1
	4	2.7e+07	76	4.7	12.4
HDEMON50	0	1.1e+08	76	13.1	8.2
	1	8.4e+07	76	11.7	9.8
	2	8.2e+07	76	10.3	8.9
	3	8.2e+07	76	10.1	8.7
	4	8.2e+07	76	10.1	8.7
LOUVAIN	0	8.7e+06	100	1.0	10.7
	1	1.4e+06	100	1.0	68.6
	2	1.0e+06	100	1.0	92.6
	3	9.8e+05	100	1.0	94.4
	4	9.8e+05	100	1.0	94.6
	5	9.8e+05	100	1.0	94.6
	6	9.8e+05	100	1.0	94.6
EGO-NETS	-	1.5e+07	69[2]	3.7	15.6
BFS	-	1.8e+07	90[1]	13.3	60.8

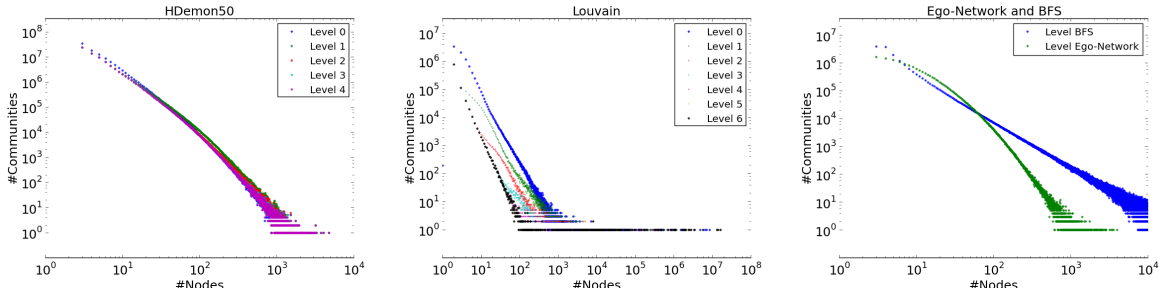
Table 7.12: Characteristics of the community sets produced by the algorithms on the analyzed dataset. For each community set we report the number of communities $\#N$, node coverage, average number of communities per node σ and average community size.

hundreds or thousands nodes, or even millions nodes as for LOUVAIN. As the hierarchical level increases, the size distributions for the HDEMON25 community sets shifts to the right, while it substantially does not in the case of HDEMON50. This is due to the overlapping parameter ψ: the smaller the parameter the higher the probability of having an edge between two meta-nodes at a given hierarchical level. Such links will increase the probability of community formation and also produce bigger communities. For LOUVAIN, as the hierarchical level increases, the curves shifts to the left: this means that the gap between huge communities and small ones increases, leading to communities that are bigger in average. The EGO-NETWORK algorithm and the BFS algorithm also produce skewed distribution, with the BFS algorithm showing the highest heterogeneity.

Community features

From the community sets produced by the four algorithms we extract a wide set of features, belonging to four main categories: *structural*, *geographical*, *formation* and *activity* features (see Table 7.13).

Structural features convey information about the topology of a social community $C = (V_C, E_C)$, where V_C and E_C are the set of nodes and edges in the community, respectively. The number of nodes N and edges M provide information about the community size. The community density $D = \frac{2M}{N(N-1)}$, i.e., the ratio between the actual links and all the possible links, indicates the level of interaction within the social group. The clustering coefficient [11] indicates how strong is the presence of triangles within the community, measuring a "all-my-friends-know-each-other" property. There are two possible definitions of the clustering coefficient: (i) the average of the local clustering coefficient of nodes $CC_{avg} = \frac{1}{N}\sum_{i=1}^{N} C_i$, where C_i is the local node clustering coefficient as defined in [11]; (ii) the transitivity ratio $CC = \frac{\text{number of closed triplets}}{\text{number of connected triples of nodes}}$. The degree assortativity A_{deg} indicates the preference for the nodes to attach to others that have the same degree [196]. Other structural features regard the level of hubbiness of a community, such as the average/maximum degree computed considering both the network links or the community links only. The diameter $d = \max_{v \in V} \epsilon(v)$ and the radius $r = \min_{v \in V} \epsilon(v)$ are respectively the maximum



(a) HDemon50 (b) Louvain (c) Ego-Network

Figure 7.17: Distribution of community size for HDemon, Louvain, Ego-Networks and BFS.

Structural features

N	number of nodes
M	number of edges
D	density
CC	global clustering
CC_{avg}	average clustering
A_{deg}	degree assortativity
deg^{C}_{max}	max degree (community links)
deg^{C}_{avg}	avg degree (community links)
deg^{all}_{max}	max degree (all links)
deg^{all}_{avg}	avg degree (all links)
T	closed triads
T_{open}	open triads
O_v	neighborhood nodes
O_e	outgoing edges
E_{dist}	num. edges with distance
d	approx. diameter
r	approx. radius
g	conductance

Community Formation features

T_f	first user arrival time
IT_{avg}	avg user inter-arrival time
IT_{std}	std of user inter-arrival time
$IT_{l,f}$	last-first inter-arrival time

Geographic features

N_s	number of countries
E_s	country entropy
S_{max}	percentage of most represented country
N_t	number of cities
E_t	city entropy
$dist_{avg}$	avg geographic distance
$dist_{max}$	max geographic distance

Activity features

Video	mean number of days of video
Chat	mean number of days of chat

Table 7.13: Description of the features extracted from the communities.

and the minimum eccentricity ϵ of any node, where the eccentricity $\epsilon(v)$ is the greatest geodesic distance between a node v and any other node in the community. They represent the linear size of a community. Finally other structural features are considered, such as the number of community neighborhoods (nodes in the global network connected to nodes in the community), the number of edges leaving the community, the number of triangles and the number of connected triples.

The **community formation** features convey information regarding the temporal appearance of nodes within the community, such as: the time of subscription to the Skype service of the first user to subscribe; the average and the standard deviation of the inter-arrival times of users, i.e., the time lapse between the subscription of a user and the next one; the inter-arrival time between the first node to subscribe and the last node who adopted the service.

Geographic features provide information about the geographic diversity of a community or, in other words, its cosmopolitan nature. The number of different countries represented gives a first estimation of the international nature of the community. The country entropy is a more refined measure, which estimates the national diversity through the Shannon entropy: $E = \sum_{c \in C} p(c) \log p(c)$, where C is the set of the countries represented in the community and $p(c)$ is the probability of the

country c to be represented in the community. We also compute the city entropy and the number of different cities represented by the community. Moreover, for the users for which we know the city name (those associated to cities with more than 5,000 users), we compute their geographic distance using the coordinates of the centers of the cities. Once computed all the available distances, we consider the average and the maximum geographic distances of each community.

Finally, the **activity** features indicate the mean level of activity performed by the community members. We extract two activity features: (i) *Chat*, the mean number of days they used the instant messaging (Chat) service; and (ii) *Video*, the mean number of days they used the Video conference service. The distributions of the chat feature for HDemon, BFS and Ego-Networks follow a normal distribution, while those of the chat feature (for Louvain) and of the video feature (for all algorithms) follow an exponential distribution. In all cases, the separation between high-engagement and low-engagement communities is less clear for higher thresholds. For the video feature, the median ranges from 3 to 3.75 (across algorithms) while the 75^{th}-percentile ranges from 6 to 7. For the chat feature, the median ranges from 5 to 5.9, while the 75^{th}-percentile ranges from 13.9 to 15.4.

Experiments

Our purpose is to use the features described above to *classify* the level of engagement of social communities, represented by the Chat and Video activity. To this purpose, we need to build a supervised classifier which assigns communities to one of several predefined categories.

The four community detection algorithms applied on the data with the specified parameter values produced a total of 19 community sets (see Table 7.12): a hierarchy of 7 levels for Louvain, two of 5 for both HDemon25 and HDemon50 and a single set for Ego-Network and BFS. Table 7.12 shows some descriptive statistics about the obtained community sets. Only Louvain guarantees the complete coverage of the nodes, since it is a partitioning algorithm: BFS and Ego-Network cover only a sample (due to the 20% node sampling strategy adopted in our experiments), while in HDemon the $\mu = 3$ parameter excludes nodes which are not involved in at least a triangle. For the Louvain community sets, we consider the hierarchical levels 0 and 6 only, which correspond to the first greedy iteration and the iteration having the maximum modularity.

Video: AUC and Accuracy

Algorithm	**Lv.**	**Scores**
HDemon25	1	**.74** (.67)
HDemon50	0	.71 (.68)
Louvain	0	.65 (.60)
Louvain	6	.63 (.59)
Ego-Nets	-	.70 (.64)
BFS	-	.67 (.62)

Chat: AUC and Accuracy

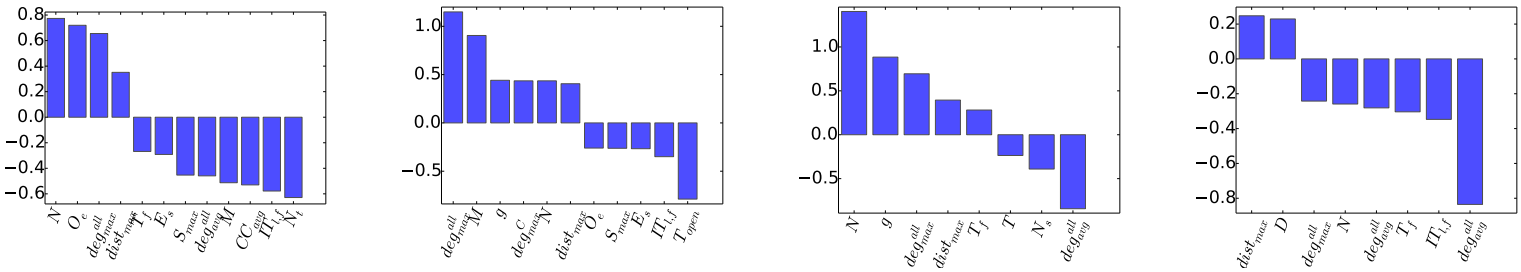
Algorithm	**Lv.**	**Scores**
HDemon25	2	**.84** (.77)
HDemon50	1	.81 (.73)
Louvain	0	.69 (.64)
Louvain	6	.65 (.60)
Ego-Nets	-	.75 (.75)
BFS	-	.81 (.72)

Table 7.14: AUC and Accuracy (within brackets) produced by the SGD method in the balanced scenario, for Video and Chat features. In bold the overall best model.

In the following, we describe the performed experiments and the results achieved: later on, we will discuss the results obtained imposing a balanced class distribution scenario and, subsequently we will test our approach on the harder, uneven class distribution one.

Balanced classes scenario

We consider two classes of user engagement for each of the two activity features (chat and video): low engagement and high engagement. To transform the two continuous activity features into discrete variables we partition the range of values through the median of their distribution. This produced, for each variable to predict, two equal-populated classes: (i) low engagement, ranging



(a) HDemon Chat (b) Ego-Network Chat (c) BFS Chat (d) Louvain Chat

Figure 7.18: Weights of the features produced by the SGD method for the community sets of HDemon (a), Ego-Network (b), BFS (c), and Louvain (d).

in the interval $[0, median]$; and (ii) high engagement, ranging in the interval $[median, 31]$.[3]

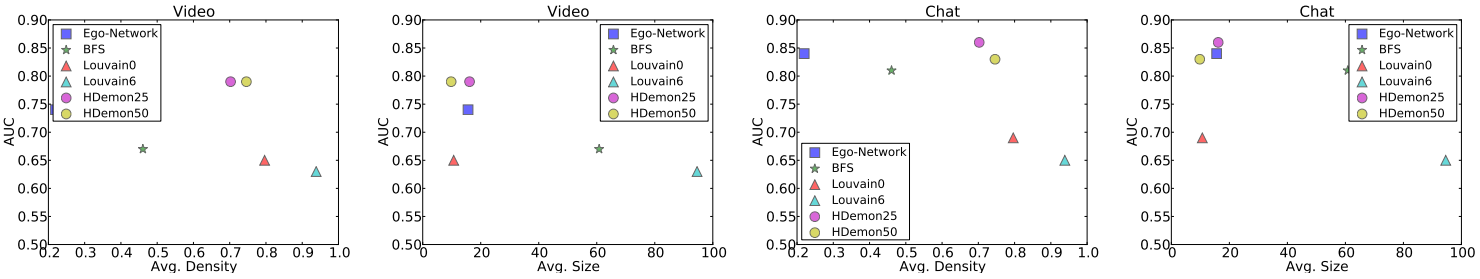
To perform classification we use Stochastic Gradient Descent (SGD) and AUC (area under the ROC curve) to evaluate their performance. The ROC curve illustrates the performance of a binary classifier, and is created by plotting the true positive rate (tpr, also called sensitivity) versus the false positive rate (fpr, also called fall-out or 1-specificity), at various threshold settings. The overall accuracy is instead the proportion of true results (both true positives and true negatives) in the population. Moreover, in a preliminary testing phase the classification step was repeated also using a Random Forest model built upon C4.5: due to the similar performance observed, the more intuitively interpretation of the obtained results and the lower execution time we decided to show only the results obtained by SGD.

Thus, we learn the SGD classifier with logistic error function [197, 198] exploiting its implementation provided by the `sklearn` Python library.[4] We execute 5 iterations, performing data shuffling before each one of them, imposing the elastic-net penalty $\alpha = 0.0001$ and $l1\text{-}ratio = 0.05$. The adoption of elastic-net penalty results in some feature weights set to zero, thus eliminating less important features. The value for the hyper-parameters is obtained by running grid-search on a smaller, random subset data from HDemon communities at level 0. We apply a five fold cross-validation for learning and testing.

Table 7.14 shows the AUC produced by the SGD method on the features extracted from the community sets produced by the four algorithms (for HDemon and Louvain only the two best performing community sets are reported). HDemon produces the best performance, both in terms of AUC and overall accuracy, for all the three activity features. Louvain, conversely, reaches a poor performance and it is outperformed by the more trivial BFS and Ego-Networks algorithms. These results suggest that the adoption of pure modularity optimization approaches, like the Louvain one, is not effective when categorizing group-based user engagement due to their resolution limit which causes the creation of huge communities [**?**]. As the level of the Louvain hierarchy increases, and hence the modularity increases, both the AUC and overall accuracy decrease. In the experiments, indeed, the first Louvain hierarchical level outperforms the last level, even though the latter has the highest modularity.

Figure 7.18 shows the features which obtain a weight value by the SGD method higher than 0.2 or lower than -0.2 (i.e., the most discriminative features for the classification process). HDemon distributes the weights in a less skewed way, while the other algorithms tend to give high importance to a limited subset of the extracted features. Moreover only a few Louvain features have a weight higher than 0.2 or lower than -0.2 (see Figure 7.18, d), confirming that a modularity approach produces communities with weak predictive power with respect to user engagement. Moreover, an interesting phenomena emerges: independently from the chosen community discovery approach, the most relevant class of features for the classification process seem to be to the *topological* one

[3] the maximum is 31 because it refers to the mean number of days per month in which that activity was performed.
[4] `http://scikit-learn.org/stable/index.html`

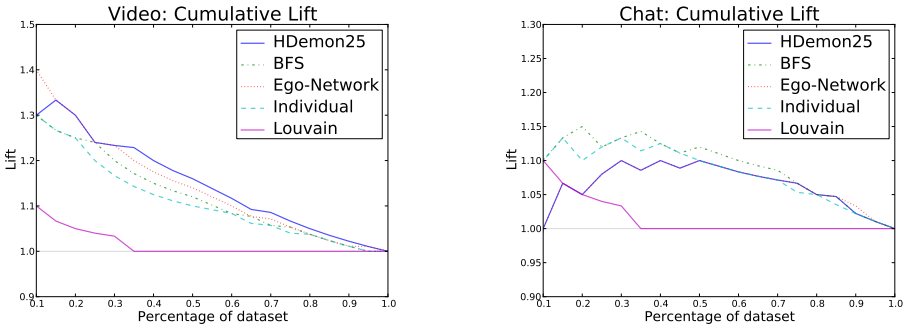


(a) AUC vs. Density: Video (b) AUC vs. Size: Video (c) AUC vs. Density: Chat (d) AUC vs. Size: Chat

Figure 7.19: Balanced Scenario: AUC vs. Avg. Density and AUC vs. Avg. Size.

(i.e., the sum of the absolute values of the SGD weights for the features belonging to such class is always greater than the same sum for *community formation* and *geographical* features combined). In particular degree, density, community size and clustering related measures often appear among the most weighted features.

Figures 7.19 shows the relationships between the average community size, the average community density and the AUC value produced by the SGD method on the community sets which reach the best performances in the balanced scenario. The best performance is obtained for the HDemon community sets, which constitute a compromise between the micro and the macro level of network granularity. When the average size of the communities is too low, as for the ego-network level, we lose information about the surroundings of nodes and do not capture the inner homophily hidden in the social context. On the other hand, when communities become too large, as in the case of communities produced by Louvain we mix together different social contexts losing definition. Communities expressing a good trade-off between size and density, as in the case of the HDemon algorithm, effectively reach the best performance in the problem of estimating user engagement.



(a) Video (b) Chat

Figure 7.20: Unbalanced scenario: Lift plot for each service.

Unbalanced classes scenario

We address also an unbalanced scenario where we use the 75^{th} percentile for the low engagement class, which thus contains the 75% of the observations, and put the remaining 25% of the observations in the high engagement class.

Table 7.15 describes the results produced by the SGD methods in the unbalanced scenario, using the same features and community discovery approaches discussed before. The baseline method for the unbalanced scenario is the majority classifier: it reaches an AUC of 0.75 by assigning each item

Video: AUC and Accuracy		
Algorithm	**Lv.**	**Scores**
HDemon25	1	**.76** (.68)
HDemon50	0	.73 (.65)
Louvain	0	.64 (.59)
Louvain	6	.61 (.58)
Ego-Nets	-	.71 (.63)
BFS	-	.68 (.61)
baseline	-	.75

Chat: AUC and Accuracy		
Algorithm	**Lv.**	**Scores**
HDemon25	2	.82 (.78)
HDemon50	3	.80 (.76)
Louvain	0	.68 (.70)
Louvain	6	.67 (.66)
Ego-Nets	-	**.83** (.79)
BFS	-	.82 (.77)
baseline	-	.75

Table 7.15: AUC and Accuracy (within brackets) produced by the SGD method in the unbalanced scenario, for the Video and Chat features. In bold the overall best model. The baseline method is the majority classifier, which reaches an AUC of 0.75 by assigning each item to the majority class (the low engagement class).

Video: Precision - Recall		
Algorithm	**Lv.**	**Scores**
HDemon25	2	**.42** (.72)
HDemon50	1	.39 (.70)
Louvain	0	.33 (.69)
Louvain	6	.33 (.67)
Ego-Nets	-	.37 (.68)
BFS	-	.35 (.71)
baseline	-	.25

Chat: Precision - Recall		
Algorithm	**Lv.**	**Scores**
HDemon25	2	.54 (.69)
HDemon50	3	.50 (.67)
Louvain	0	.40 (.41)
Louvain	6	.44 (.33)
Ego-Nets	-	**.57** (.68)
BFS	-	.52 (.71)
baseline	-	.25

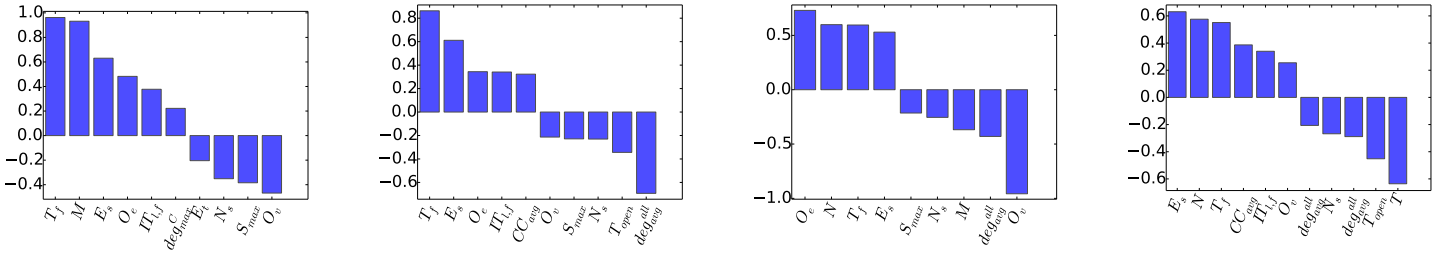
Table 7.16: Precision and Recall (within brackets) produced by the SGD model for the Video and Chat features in the unbalanced scenario. In bold the overal best model. Having used the 75^{th} percentile to discriminate the class labels the Precision baseline w.r.t. the positive class is .25.

to the majority class (the low engagement class). We observe that, regardless the community set used, the SGD method (as well as Random Forest) is not able to improve significantly the baseline classifier for Video. Conversely, the results obtained for the Chat feature by SGD outperform the baseline when we adopt HDemon, Ego-Networks and Louvain community sets, reaching an AUC of 0.83.

In order to provide additional insights on the models built with the adoption of the different CD algorithms, we also compute the precision and recall measures with respect to the minority class (see Table 7.16). Looking at these measures enable us to understand which are the advantage in using SGD to identify correctly instances of the less predictable class. Moreover, due to the skewed right side of the Video and Chat distributions we can observe how choosing the 75^{th} percentile lead to a very difficult classification setup: the instances belonging to the minority class often represent outliers having very few examples from which the classifier can learn the model.

Here, the baseline is the minority classifier which reaches a precision of 25% by assigning each community item to the minority class (the high engagement one). We observe that the SGD method outperforms the baseline classifier on all the community sets (reaching values in the range [.33, .57]). HDemon and Ego-Networks are the community sets which lead to the best precision, on the Video features and the Chat feature respectively.

In order to measure the effectiveness of SGD we report the Lift chart which shows the ratio between the results obtained with the built model and the ones obtained by a random classifier. The charts in Figure 7.20 are visual aids for measuring SGD's performance on the community sets: the greater the area between the lift curve and the baseline, the better the model. We observe that HDemon perform better than the competitors for the video features. For the chat features, the community sets produced by the three naive algorithm win against the other two CD algorithms. For all the three activity features Louvain reaches the worst performance, as in the balanced scenario.



(a) HDemon Video (b) Ego-Network Video (c) BFS Video (d) Louvain Video

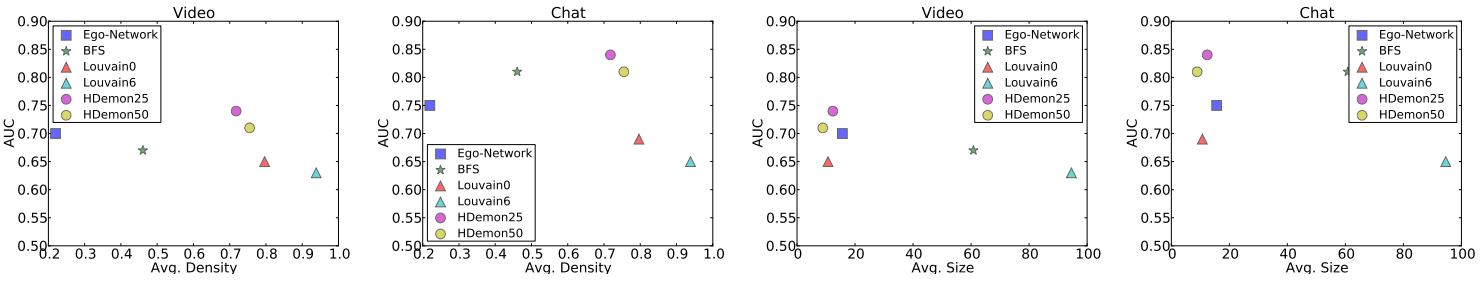
Figure 7.21: Unbalanced scenario: Stochastic Gradient Descent features weights for all the community sets. HDemon (a), Ego-Network (b), BFS (c), and Louvain (d).

As done for the balanced scenario, in Figure 7.21 we report for each CD the features having weight greater than 0.2 or lower than -0.2. Conversely from the results presented in the previous section, where topological features alway show the higher relative importance for the classification process, in this scenario we observe how *community formation* and *geographical* features are the ones which ensure greater descriptive power. As previously observed the minority class identified by a 75^{th} percentile split is mostly composed by particular, rare, community instances. This obviously affect the relative importance of temporal and geographical informations: the results seems to suggest that the more a community is active the more significative are its geographical and temporal bounds.

Finally in Figure 7.22 we show the relationships between the average community size, the average community density and the AUC value produced by the SGD method on the community sets which reach the best performances in the unbalanced scenario. We can observe how, in this settings, the algorithms granting communities having on average small sizes and high density are the ones that assure the construction of SGD models reaching higher AUC. In particular HDemon in both its instantiation seems to outperform the other approaches.

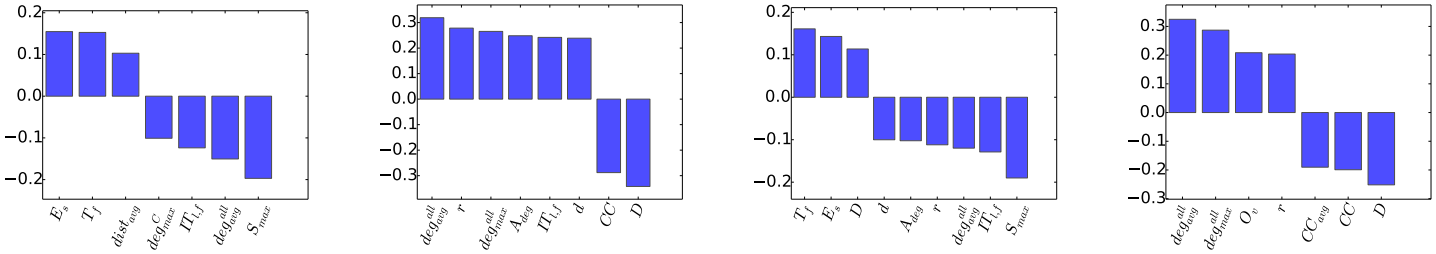
Community Characterization

From our analysis emerged a well defined trend: among the compared methodologies, HDemon is able, both in balanced and unbalanced scenarios, to better bound homophily and thus to extract communities whose analysis guarantee useful insights on the product engagement level. For this reason, starting from the communities extracted by such bottom-up overlapping approach we computed the Pearson correlation for all the defined features against the final class label (high/low engagement).



(a) AUC vs. Density: Video (b) AUC vs. Density: Chat (c) AUC vs. Size: Video (d) AUC vs. Size: Chat

Figure 7.22: Unbalanced Scenario: AUC vs. Avg. Density and AUC vs. Avg. Size.



(a) Video median (b) Chat median (c) Video 75^{th} percentile (d) Chat 75^{th} percentile

Figure 7.23: Most relevant Pearson correlations between community feature values and target class (high/low activity) for HDemon. In (a-b) are shown the indexes for the balanced class scenario while in (c-d) for the 75^{th} percentile split.

As shown in Figure 7.23(a), when splitting the Video engagement using the 50^{th} percentile we are able to identify as highly active communities the ones having high country entropy E_s as well as high geographic distance among its users $dist_{avg}$ and whose formation is recent (i.e., whose first user has joined the network recently, T_f, as well as the last one, $IT_{l,f}$.). Moreover, Video active communities tend to be composed by users having on average low degree as shown by deg^{all}_{avg} and deg^{C}_{max}. Conversely, looking at Figure 7.23(b) we can notice that communities which exhibit high Chat engagement can be described by persistent structures (i.e., social groups for which the inter-arrival time $IT_{l,f}$ from the first to the last user is high), composed by users showing almost the same connectivity (in particular having high degree) and sparse social connections (low clustering coefficient CC, low density D and high radius).
Moreover, we calculate the same correlations for the 75^{th} percentile split: conversely from the new results for the Chat engagement (Figure 7.23(d)) which do not differ significantly from the ones discussed for the balanced scenario, in this settings the highly active Video communities show new peculiarities. In Figure 7.23(c) we can observe how the level of engagement inversely correlates with the community radius (and diameter) and directly with the density. These variations describe highly active Video communities as a specific and homogeneous sub class composed by small and dense network structures composed by users who live in different countries (high geographical entropy E_s).

Discussion

As previously done for the Quantification problem in 7.2, in this work we have studied the relation between homophily and community discovery. Coherently with our previous results a clear pattern emerges from the conducted analysis: CD algorithms are able to bound homophilic behaviors only if the their community definition is able to produce dense, small size communities. In particular, using information extracted by communities (both semantic and structural) we were able to learn very accurate classifiers for the Skype products. We first produced several community sets from the global Skype network by applying on the data different community detection algorithms. We then extracted from each community topological and geographical features and learned classification models to predict the level of usage for the video and chat services provided by the VOIP provider. Our results showed that, both in balanced and unbalanced classification scenarios, algorithms producing overlapping micro-communities like HDemon reach the best performances. Conversely, modularity-based approach like Louvain are not able to guarantee good performance and are often outperformed by simpler clustering strategies such as Ego-Networks and BFS. Moreover, in order to give some insights on the obtained classification, we provided a description for low/high engaged communities identified by HDemon through the analysis of the correlations between their activity level and the values of their features. Our results could be further improved by two key

properties which are not present in the Skype dataset: the strength of the ties between the users, and information about the dynamics of user profiles and network links. On one side, the strength of a tie quantifies the degree of interaction between two individuals, allowing to understand at what extent the level of interactions inside a community is linked to its service engagement. On the other side, temporal information about the appearance of links or the geographical location of users will allow us to investigate how the network and community structure, and hence the service engagement, change in time and after the migration of users from a country to another.

Part III

Social Dynamics: Networks through Time

Chapter 8

Modeling Individual Dynamics

> Change will not come if we wait for some other person or some other time. We are the ones we've been waiting for. We are the change that we seek.
>
> — *Barack Obama*

Moving from the results introduced in Part II in this chapter we tackle the problem of predicting new links. As discussed in 4.1.2, several families of methodologies were proposed to address this very complex task. Our first work on this subject, introduced in 8.1, provides and evaluates a handful of unsupervised methods that exploit multidimensionality and temporal annotation in order to solve the LP problem in the specific context of multidimensional networks.
In 8.2 we reformulate the classic definition of the Link Prediction task in order to describe a more challenging problem: Interaction Prediction (henceforth, IP). In this scenario we address the more general task of forecasting new interactions in a highly dynamic social scenario. In order to achieve high predictive performances, we propose a mining procedure which takes advantage of our results in Community Discovery (already discussed in 7.1) as well as time series forecasting and supervised learning approaches.

8.1 Unsupervised Link Prediction[1]

Moving from the analysis of multidimensional networks proposed in the previous part of this thesis, here we introduce an extension of the Link Prediction problem. Following the approach of a large family of studies based on structural properties of the network such as, for example, Common Neighbors [62], Preferential Attachment [60] or Adamic-Adar [61], we will introduce several measures able to exploit the knowledge that can be extracted from the multiplex networks in order to predict those links that will appear in each specific dimensions. In our vision, the evolution of a multidimensional network depends on three factors:

i) the underlying theoretical model of node interactions (e.g., nodes with high degree tend to attract more connections);

ii) the interplay among dimensions (e.g., links may form in a specific dimension with a higher likelihood);

iii) the complete temporal history of a link (e.g., links always present during the network history may be more likely to appear in the future).

In oder to reflect this, we build unsupervised predictors which combine the contribution of three basic measurements used in conjunction: (i) multidimensional versions of Common Neighbors and Adamic-Adar, (ii) global measures capturing the interplay of multiple dimensions at different levels, and (iii) measures based on the complete history of the presence of a link within a network.

Multidimensional Networks: Temporal extension

As shown in 6.1, we adopt a *multigraph* to model multidimensional networks and their properties. Moreover, we extend the previously introduced model in order to express temporal annotation on the edges. Since we do not consider node labels, hereafter we use *edge-labeled undirected multigraphs*, denoted by a tuple $\mathcal{G} = (V, E, L, T, \tau)$ where: V is a set of nodes; L is a set of labels; E is a set of labeled edges, i.e., the set of triples (u, v, d) where $u, v \in V$ are nodes and $d \in L$ is a label; T is a set of timestamps; $\tau : E \rightarrow \mathcal{P}(T)$ is a function returning the set of timestamps of presence of a given edge (and $\mathcal{P}(T)$ denotes the power set of T). Where we are not interested in the temporal history of an edge, we refer to it by simply using a triple (u, v, d). Whenever, in turns, we need to specify the temporal information, we use the pair $((u, v, d), \tau(u, v, d))$. Moreover, if we write $(u, v, d) \in E$ we assume: $\tau(u, v, d) \neq \emptyset$, i.e., there exist at least one timestamp t in which the edge (u, v, d) is present, or, in other words, in which the dimension d connects u and v. Hereafter, we omit this note in the definitions of our structural measures when this is not needed.

Also, we use the term *dimension* to indicate *label*, and we say that a node *belongs to* or *appears in* a given dimension d if there is at least one edge labeled with d adjacent to it. We also say that an edge *belongs to* or *appears in* a dimension d if its label is d. We assume that given a pair of nodes $u, v \in V$ and a label $d \in L$ only one edge (u, v, d) may exist. Thus, each pair of nodes in $\mathcal{G}$ can be connected by at most $|L|$ possible edges. Hereafter $\mathcal{P}(L)$ denotes the power set of L.

8.1.1 Multidimensional Link Prediction

Given a pair of nodes in an evolving network, the literature on monodimensional network analysis defines Link Prediction as the problem of estimating the likelihood that an edge will form between two nodes. There can be several ways to reformulate it in the multidimensional setting. For example, the classical definition may be preserved as it is, disregarding the dimensions, only focusing on new connections between any two nodes. Another possible way is to specify a set of dimensions for which we want to estimate the likelihood. A more specific formulation, that we use in the rest of this work, require to estimate the likelihood that an edge will form between two nodes in a specific

[1] G. Rossetti, M. Berlingerio, and F. Giannotti, "Scalable link prediction on multidimensional networks", in IEEE ICDM, 2011.

dimension. That is, we add an additional parameter to the classical definition. More formally we define:

Definition 15 (Multidimensional Link Prediction) *Given a multidimensional network modeled as a multigraph $\mathcal{G} = (V, E, L, T, \tau)$, the Multidimensional Link Prediction problem (from now on, MLP) requires to return a function* score $: V \times V \times L \to [0, +\infty[$ *of scores measuring the likelihood that any two pairs of nodes will connect in a specific dimension, in the future.*

Local Measures: Neighborhood

Besides the classic definition of neighborhood, we adopt as connectivity measures the ones defined in 6.1: in particular the $Neighbor_{Xor}$ variant. Moreover, when computing the connectivity measures we assume: $\tau(u, v, d) \neq \emptyset$, thus we omit the temporal information in our measures.

While the classic formulation of the function $Neighbors$ might be used directly into the formulas of Common Neighbors and Adamic-Adar we will use its Xor variant in order to find a better fit for the multidimensional settings which is able to capture the interplay among the dimensions w.r.t. the exclusivity of connections.

Global Measures: Dimension Connectivity

While the Neighbor related measures defined in 6.1 are local to nodes, here we define four novel global measures. In particular, we introduce the *Dimension Connectivity* and *Average Correlation* measures on both the sets of nodes and edges.

Definition 16 (Node Dimension Connectivity) *Let $d \in L$ be a dimension of a network $\mathcal{G} = (V, E, L, T, \tau)$. The function $NDC : L \to [0, 1]$ defined as*

$$NDC(d) = \frac{\mid \{u \in V \mid \exists v \in V : (u, v, d) \in E\} \mid}{|V|} \tag{8.1}$$

computes the ratio of nodes of the network that belong to the dimension d.

Definition 17 (Edge Dimension Connectivity) *Let $d \in L$ be a dimension of a network $\mathcal{G} = (V, E, L, T, \tau)$. The function $EDC : L \to [0, 1]$ defined as*

$$EDC(d) = \frac{|\{(u, v, d) \in E | u, v \in V\}|}{|E|} \tag{8.2}$$

computes the ratio of edges of the network labeled with the dimension d.

While the two measures above regard the importance that a single dimension has w.r.t. the connectivity of the network, we now define other two measures aimed at capturing the global interplay among dimensions by looking at their average correlation.

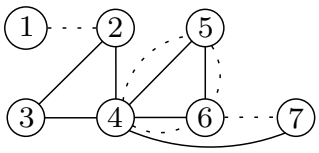
Definition 18 (Average Node Correlation) *Let $d \in L$ be a dimension of a network $\mathcal{G} = (V, E, L, T, \tau)$. The function $ANC : L \to [1/|L|, 1]$ is defined as*

$$ANC(d) = \frac{\sum_{d' \in L} NJaccard(d, d')}{|L|} \tag{8.3}$$

where $NJaccard(d, d')$ is the Jaccard correlation index on the node sets $\frac{|N(d) \cap N(d')|}{|N(d) \cup N(d')|}$, where $N(\bar{d}) = \{u \mid \exists\ (u, v, \bar{d}) \in E\}$. It computes the average node correlation of a dimension with all the others.

Definition 19 (Average Edge Correlation) *Let $d \in L$ be a dimension of a network $\mathcal{G} = (V, E, L, T, \tau)$. The function $AEC : L \to [1/|L|, 1]$ is defined as*

$$AEC(d) = \frac{\sum_{d' \in L} EJaccard(d, d')}{|L|} \tag{8.4}$$



$\tau(1,2,2) = \{1,3\}$ $\tau(2,3,1) = \{2,3\}$
$\tau(2,4,1) = \{1\}$ $\tau(3,4,1) = \{3\}$
$\tau(4,5,1) = \{2,3\}$ $\tau(4,5,2) = \{2,3\}$
$\tau(4,6,1) = \{3\}$ $\tau(4,6,2) = \{2,3\}$
$\tau(4,7,1) = \{1,2\}$ $\tau(5,6,1) = \{1,2,3\}$
$\tau(5,6,2) = \{1,3\}$ $\tau(6,7,2) = \{1,2,3\}$

Figure 8.1: Toy example: Solid line is dimension 1, dashed is dimension 2.

where $EJaccard(d,d')$ is the Jaccard correlation index on the edge sets $\frac{|E(d)\cap E(d')|}{|E(d)\cup E(d')|}$, where $E(\bar{d}) = \{(u,v) \mid \exists(u,v,\bar{d}) \in E\}$. It computes the average edge correlation of a dimension with all the others.

Example 2 *In Figure 8.1 the EDC of dimension d_1 (solid line) is $7/12$ since it has 7 edges out of the 12 total edges of the network, while the EDC of d_2 (dashed line) is $5/12$. The NDC for d_1 is $5/7$ and NDC for d_2 is $6/7$. The AEC of d_1 is $(1+3/12)/2 = 0.625$. For the same dimension, ANC is $(1+5/7)/2 = 0.857$.*

Temporal Measures

Besides the analysis of the multidimensional structure at both the local and global levels, we also want to take into account the complete temporal history of each edge belonging to the network. In order to do so, we define four different measures. Thus, here we make use of the τ function.

The first measure simply counts the number of temporal snapshots in which an edge is present in a dimension:

Definition 20 (Frequency) *Let $(u,v,d) \in E$ be an edge of a network $\mathcal{G} = (V,E,L,T,\tau)$. The function $Freq : E \to [1,|T|]$ defined as*

$$Freq(u,v,d) = \mid \tau(u,v,d) \mid \tag{8.5}$$

computes the frequency of an edge in terms of the number of temporal snapshots in which it appears.

We can aggregate the above by dimensions, counting the number of snapshots in which a pair of nodes is connected:

Definition 21 (Over All Frequency) *Let (u,v) be two nodes in V in a network $\mathcal{G} = (V,E,L,T,\tau)$. We define $OAFreq : V \times V \to [1, |L| \times |T|]$ as:*

$$OAFreq(u,v) = \left| \bigcup_{\{d \in L | (u,v,d) \in E\}} \tau(u,v,d) \right| \tag{8.6}$$

As time has a natural ordering, we may want to be able to give more (or less) importance to more recent interactions when predicting new ones. To this end, we define two weighted measures on the temporal history of an edge:

Definition 22 (Weighted Presence) *Let $(u,v,d) \in E$ be an edge of a network $\mathcal{G} = (V,E,L,T,\tau)$. The function $WPres : E \to [1,+\infty[$ is defined as*

$$WPres(u,v,d) = \sum_{\{t \in \tau(u,v,d)\}} w_t \tag{8.7}$$

where w_t is the weight of the temporal snapshot t. For simplicity, given the temporal ordering, we assume $w_{t_i} = i$.

As done above, we can also aggregate $WPres$ by dimensions:

Definition 23 (Over All Weighted Presence) *Let (u, v) be two nodes in V in a network $\mathcal{G} = (V, E, L, T, \tau)$. The function $OAWPres : V \times V \rightarrow [1, +\infty[$ defined as*

$$OAWPres(u, v) = \sum_{\{d|(u,v,d)\in E\}} WPres(u, v, d) \tag{8.8}$$

Example 3 *In the toy example in Figure 8.1, where we reported also the complete history of each edge in the table, we have: Freq(4,5,1)=2; OAFreq(4,5)=4; WPres(4,5,1)=5; OAWPres(4,5)=10.*

Predictive Models based on structural analysis

We now present several possible solutions for MLP, introducing a list of functions to use as scores. It is clear how, in analogy with the LP problem in the monodimensional case, there can be a taxonomy of solutions, divided in supervised or unsupervised approaches, based on structural analysis or on the extraction of frequent patterns of evolution, based on statistical analysis of temporal series, and so on. In the rest of this section we present solutions based on the structural analysis of the network. We start from the multidimensional reformulation of two classical approaches based on neighborhood (Common Neighbors and Adamic-Adar), then we introduce other measures to be taken into account in the final list of scoring functions. Our resulting solutions are then combinations of supervised and unsupervised approaches, aimed at capturing all the possible strong and weak signals of the non-trivial interplay of multidimensionality and temporal evolution.

All the available theoretical basic bricks previously defined can now be combined to build our set of predictors for MLP. For convenience, in this section we use the notation $N(\circ, \bullet)$ for $Neighbors(\circ, \bullet)$, and, in analogy, $N_{XOR}(\circ, \bullet)$ for $Neighbors_{XOR}(\circ, \bullet)$.

Base predictors

We wanted to have basic predictors for our experiments, and we choose Common Neighbors [62] and Adamic-Adar [61], as they are among the best w.r.t. predictive performances, as highlighted by [63]. We can introduce a multidimensional version of them by using our function Neighbors:

Definition 24 (Multidimensional Common Neighbors) *Let $\mathcal{G} = (V, E, L, T, \tau)$ be a network and $(u, v, d) \notin E$ be a candidate future edge. We define:*

$$M\text{-}CN(u, v, d) = \mid N(u, d) \cap N(v, d) \mid \tag{8.9}$$

Hereafter, we often use M-CN to refer to this predictor.

Definition 25 (Multidimensional Adamic Adar) *Let $\mathcal{G} = (V, E, L, T, \tau)$ be a network and $(u, v, d) \notin E$ be a candidate future edge. We define:*

$$M\text{-}AA(u, v, d) = \sum_{z\in\{N(u,d)\cap N(v,d)\}} \frac{1}{log(|N(z, d)|)} \tag{8.10}$$

Hereafter, we often use M-AA to refer to this predictor.

In the following, instead, we replace Neighbors with Neighbors$_{XOR}$, by following the intuition that more sophisticated multidimensional information may lead to better predictive performance. As we will observe in our experimental results, this intuition was proved to be incorrect in the networks used.

Definition 26 (Multidimensional Common Neighbors$_{XOR}$) *Let $\mathcal{G} = (V, E, L, T, \tau)$ be a network and $(u, v, d) \notin E$ be a candidate future edge. We define:*

$$M\text{-}CN_{XOR}(u, v, d) = \mid N_{XOR}(u, d) \cap N_{XOR}(v, d) \mid \tag{8.11}$$

Base	Multidim. Measure	Temporal Measure	Base	Multidim. Measure	Temporal Measure
M-AA			M-CN		
M-AA	NDC		M-CN	NDC	
M-AA	EDC		M-CN	EDC	
M-AA	AEC		M-CN	EC	
M-AA	ANC		M-CN	NC	
M-AA		Freq	M-CN		Freq
M-AA		OAFreq	M-CN		OAFreq
M-AA		WPres	M-CN		WPres
M-AA		OAWpres	M-CN		OAWPres
M-AA	AEC	WPres	M-CN	AEC	WPres
M-AA	AEC	OAWPres	M-CN	AEC	OAWPres
M-AA	ANC	WPres	M-CN	ANC	WPres
M-AA	ANC	OAWPres	M-CN	ANC	OAWPres

Table 8.1: Taxonomy of the proposed approaches (non-XOR versions)

Definition 27 (Multidimensional Adamic Adar$_{XOR}$) *Let $\mathcal{G} = (V, E, L, T, \tau)$ be a network and $(u, v, d) \notin E$ be a candidate future edge. We define:*

$$M\text{-}AA_{XOR}(u, v, d) = \sum_{z \in \{N_{XOR}(u,d) \cap N_{XOR}(v,d)\}} \frac{1}{log(|N_{XOR}(z, d)|)} \tag{8.12}$$

Observations on the scoring families

- In principle, it is possible to define several scores on the basis of the multidimensional measures presented above. For example, it is possible to multiply the $Neighbors_{XOR}$ of two nodes in one dimension to obtain a score, ending up with a Preferential-Attachment like model [60]. We tried several combinations, but, due to extremely poor predictive performances, as tested during our experimental stage, we do not report their definition. According to our experiments, in fact, the multidimensional information gathered by our measures in the networks used is not enough to predict new edges. This negative result is analog to the one obtained by the authors of [73], who reported that their supervised model was not performing well when used alone for prediction. In analogy with their strategy, we tried then to combine the information learned from the data with unsupervised model.

- It is possible to define temporal scores based on modifications of the above measures. We tried a few of them but, in analogy with the multidimensional scores, their predictive power when used alone was very poor on our networks.

- Finally, we can define a scoring function by combining all the basic bricks presented in our theory. In particular, we can aggregate the information provided by the baseline models with the information provided by the multidimensional measures or the temporal ones. This is exactly the line followed in [73], where the authors combine the information provided by the model defined by the complete set of frequent evolution rules mined from the network with the information provided by the baseline models. In analogy with their paper, we tried several combinations of our proposed measures. Table 8.1 shows the non-XOR versions of all the solutions we tested. Each line represents which basic bricks we used for building one scoring function, for a total of 26 predictors. The basic bricks were combined by multiplying their scores. Clearly, other aggregates or combinations are possible and we tried some of them, but, due to poor predictive power, here we only report the best ones.

Dataset	$\|V\|$				$\|E\|$				Neighbors			
	min	max	avg	global	min	max	avg	global	min	max	avg	gl. avg
DBLP train.	378	3 891	1 718.3	33 329	560	7 792	3 418.4	95 727	14	77	35.5	5.07
DBLP test	26	1 126	404.8	8 507	13	1 963	672.9	17 496	1	24	12.9	3.87
IMDb train.	9	9 219	1 581.4	12 146	36	310 811	39 568.3	989 208	8	885	228.1	62.84
IMDb test	3	2 181	354.1	2 844	3	36 658	4 676.4	116 910	2	161	61.7	31.43

Table 8.2: Basic statistics for our networks

Experiments

In order to better understand how the proposed approaches behave we report the results obtained by applying them on real complex networks. The predictive performance is measured via ROC (Receiver Operating Characteristic) curves computed on the results of the predictors. We use ROC curves instead of Precision/Recall plots for their better comparability among different networks and predictors. Moreover, as shown in [199] both curves lie in isomorphic spaces.

We choose to use for testing purposes two networks coming from different real world sources: the bibliographic database DBLP[2], from which we extracted a co-authorship network, and the movie database IMDb[3], from which we extracted a collaboration network. More in details, we built the following two networks:

- **DBLP.** We extracted author-author relationships if two authors collaborated at least in one paper. The dimensions are defined as the venues in which the paper was published. We took only the publications in the most important 28 conferences in computer science, which include VLDB, SIGKDD, WWW, AAAI and more. For the training set we narrowed the temporal span to the 1999-2008 years and chose year 2009 as test set.

- **IMDb.** We extracted a collaboration network of the actors involved in Indian movie productions. Two actor (nodes) are connected by an edge if they took part in at least one movie together in a given year: as training set we considered the years from 1999 to 2008 and the year 2009 as test set. To introduce multidimensionality we took care, for each actor-actor edge, of the genres of the movie, ending up with 25 different dimensions.

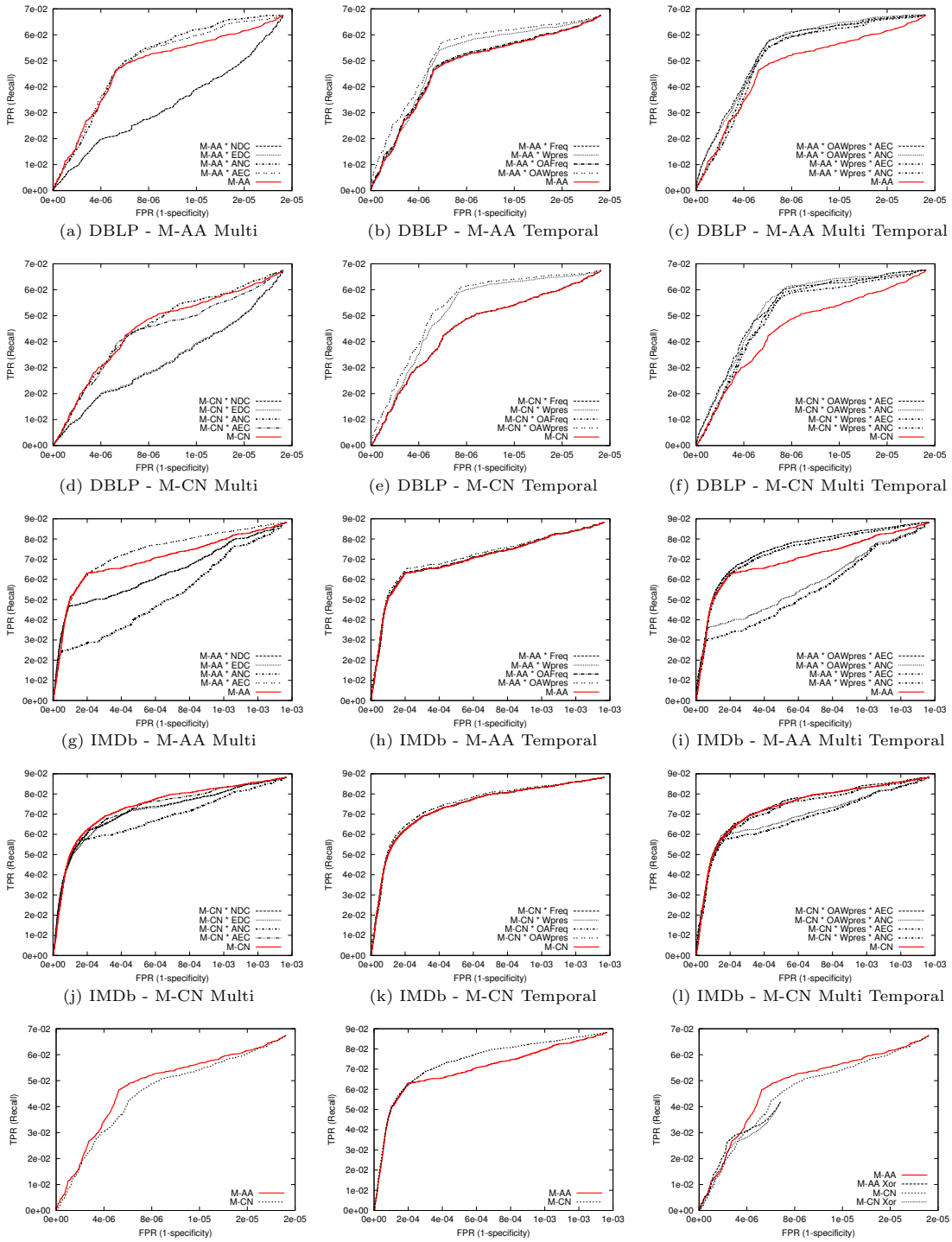
Table 8.2 summarizes, for each network and set considered, the number of nodes, edges, and neighbors, and reports for each of them min, max and average computed over the different dimensions as well as their global values computed disregarding the multidimensional information (where "gl. avg" is the average degree).

Evaluation of the results

We now want to give a quantitative evaluation of the results. We measure how well M-CN, M-AA and their XOR versions perform, how the two versions of them compare, how much their predictive power can be improved by multidimensional or temporal information, and we want to see if there are global predictors that globally outperform all the others.

We applied all the scoring functions as reported in Table 8.1 to our networks. In Figure 8.2 we report the ROC curves computed on a significative selection of results. Figure 8.2 reports in the first two rows the ROC curves computed in DBLP by using M-CN and M-AA, multiplied by multidimensional information (first column), temporal information (second column) or both of them (last column). The second row report the same, for IMDb. The last row of the figure reports different sets of plots. In the first two column of Figure 8.2 we show the comparison between the M-AA and M-CN on DBLP and IMDb, respectively, while in the third we group all the four multidimensional base predictors on the DBLP network. In Figure 8.3, we report the

[2]http://dblp.uni-trier.de
[3]http://www.imdb.com



(a) DBLP - M-AA Multi (b) DBLP - M-AA Temporal (c) DBLP - M-AA Multi Temporal

(d) DBLP - M-CN Multi (e) DBLP - M-CN Temporal (f) DBLP - M-CN Multi Temporal

(g) IMDb - M-AA Multi (h) IMDb - M-AA Temporal (i) IMDb - M-AA Multi Temporal

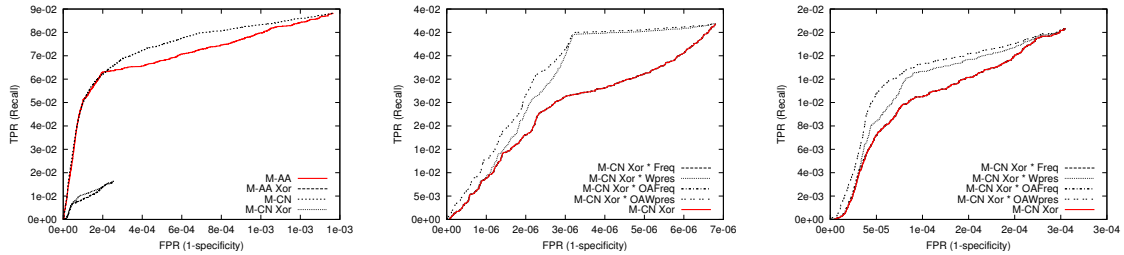
(j) IMDb - M-CN Multi (k) IMDb - M-CN Temporal (l) IMDb - M-CN Multi Temporal

(m) DBLP
M-AA, M-CN

(n) IMDb
M-AA, M-CN

(o) DBLP
M-AA, M-AA-Xor, M-CN, M-CN-Xor

Figure 8.2: ROC curves computed on the predictors based on Neighbors



(a) IMDb
M-AA, M-AA-Xor, M-CN, M-CN-Xor

(b) DBLP
M-CN-Xor Temporal

(c) IMDb
M-CN-Xor Temporal

Figure 8.3: ROC curves computed on the predictors based on Neighbors$_{XOR}$

comparison between all the four multidimensional base predictors on IMDb, and two examples of the performances given by the predictors based on $Neighbors_{XOR}$.

First, by comparing figures 8.2 and 8.3, we must observe a negative result: the XOR variant of the Neighbors function is destroying part of the information about the neighbors. This can be seen by the scale on the y axis of all the plots in Figure 8.3 (there, we show only the best results obtained by the XOR versions). Due to the definition of our measures, the XOR is reducing the number of total predictions issued. This is very clear from the plots in figures 8.2(o) and 8.3(a), that compare the normal and XOR versions of the basic predictors for both the networks.

Second, the temporal scores used as multiplier for the base predictors are able to restore [4] part of the predictive power lost with the XOR, which can be additionally recovered by multiplying also by the multidimensional measures. However, the XOR based prediction is globally very poor compared to the normal one.

Third, consider the plots in figures 8.2(m) and 8.2(n). Here is shown the comparison between M-AA and M-CN for the two networks. As we see, while in DBLP the global better performance of the Adamic-Adar predictor is validated, this is not true for IMDb, where M-CN is performing better. A possible explanation of this might be found in the structure of the two networks. In addition to the global more dense structure of IMDb, we also note that this network tends to have more cliques, that are also larger, w.r.t. DBLP, given that one movie usually joins together more persons than one scientific paper. In this scenario, the prevalence of the Adamic-Adar intuition (more importance to rarer neighbors) over the Common Neighbors one (higher score to nodes with more neighbors in common) seems to lose its strength. In turns, M-CN for IMDb seems to be difficult to boost by means of multidimensional or temporal information, as we can see in the fourth row of Figure 8.2, that, on the other hand, destroy part of the predictive power of M-CN.

Next, consider the first two rows of Figure 8.2. In DBLP, the predictive power of both M-CN and M-AA can be boosted by adding multidimensional or temporal information, with an even more powerful conjunction (see (c) and (f)).

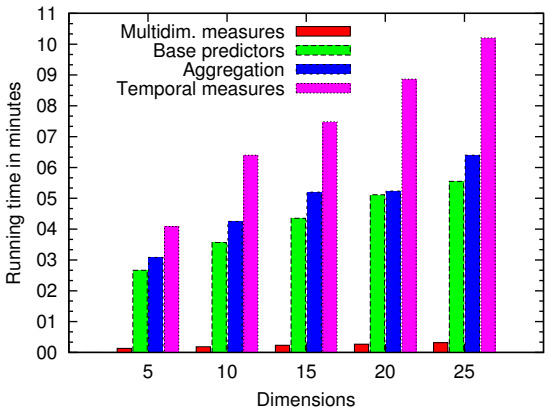
Regarding which multidimensional or temporal measures are able to help the prediction, we see that: for the first, ANC and AEC globally tend to add predictive power (especially ANC), while NDC and EDC globally lower the precision; for the second, it is clear from all the plots that the weighted version of all the measures is more accurate in capturing the temporal information, and that the OverAll versions of the measures behave better than the normal ones.

Globally speaking, the best predictors in the networks analyzed is the one built upon the conjunction of OAWpres, ANC, and one of M-CN and M-AA.

Comparison with the random predictor

In analogy with previous works [63], we also compare the performance of our predictors with the random one, used as baseline. The performance is here identified with the precision of the

[4] Not in terms of number of predictions, but in terms of their precision.



Dimensions	$\|V\|$	$\|E\|$
5	9 927	378 675
10	10 987	563 497
15	11 573	711 097
20	11 716	843 506
25	12 146	989 208

Figure 8.4: Running times and statistics for the analyzed different IMDB subnetworks.

predictive model, which for the random predictor can be calculated as:

$$Precision_{random} = \frac{|E_{test}|}{|L| \times \frac{|V|(|V|-1)}{2}} \tag{8.13}$$

Given that all the introduced predictors are based on common neighborhood, they all output the same set of predicted links, even if the scores might be different. For this reason, we can calculate a single value of performance for all the proposed predictors:

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} \tag{8.14}$$

The boost of the performance for the networks analyzed, computed as the ratio $\frac{Precision}{Precision_{random}}$, was 35,150.4 for DBLP, and 7.1 for IMDb. As we see, the gain is much higher for DBLP, which is a much sparser network (see Table 8.2).

Scalability

The last open question regards the degree of scalability of our approach. In order to answer it, we built a few network with different node and edge sizes. In particular, we took the training set of IMDb (the largest network), and produced 5 different subnetworks. We started taking the nodes and edges belonging to only 5 dimensions, producing a first small network, then we added 5 dimensions (with the corresponding nodes and edges) at each step, until the entire network was added. Table 8.4 summarizes the basic statistics (only number of nodes and edges) of the subnetworks produced in this way. We decided to divide the 25 dimensions in such a way that the number of edges would have increased (almost) uniformly. On these networks, we computed all the measures, the predictors, and the aggregations as reported above. Figure 8.4 reports the running times (in minutes) for the experiments. Since we had many aggregations, instead of reporting the total computing time, we split it into four steps: computing the multidimensional measures (first bar in every block of four); computing the multidimensional base predictors M-CN, M-CN$_{XOR}$, M-AA and M-AA$_{XOR}$ (second bar); computing all the aggregations (third bar); and computing the temporal measures. As we see the running time grows linearly with the number of edges, with a maximum time of 30 minutes. According to their definition, and to this empirical evaluation, our proposed predictors are scalable, and the required computing time grows linearly with the number of edges.

Discussion

We have formulated the Multidimensional Link Prediction problem, and introduced different classes of scalable predictors aiming at capturing the underlying model of node interactions, the multi-

dimensional information and the complete temporal history of a link in the network. We have shown that it is possible to predict new links in multiplex networks, and our results confirm the ones provided by the literature of unsupervised monodimensional link prediction: although unsupervised models such as the Adamic-Adar or the Common Neighbors have a high influence in the evolution of a network, their accuracy as predictors may be boosted by combining them with the analysis of peculiar structural (i.e., the ones introduced by multidimensional measures) and temporal properties.

8.2 Superivsed Interaction Prediction[1]

In 8.1 we have observed how, even applying simple unsupervised approaches, we were able to achieve interesting performances in predicting new edges in multiplex networks. Moreover, exploit topological information along with additional semantic carried by nodes and edges (i.e., dimensionality and temporal patterns) appears to be the road to follow in order to increase the accuracy of well-known link prediction methodologies. This observation has lead to the proliferation of works (as [200]) which propose supervised strategies able to tackle from a data mining perspective this complex problem. As we have already discussed, the classic formulation of the *Link Prediction* problem involves the use of the observed network status to predict new edges that are likely to appear in the future as well as to unveil hidden connections among existing nodes. However, graph structures are often used to describe rapid-scale human dynamics: social interactions, call graphs, buyer-seller scenarios and scientific collaborations are only few examples. For this reason, here our aim is to exploit the temporal information carried by the appearance and disappearance of edges in a fully dynamic context to overcome the limitations imposed by the analysis of a static scenario when making predictions.
To model rapid scale dynamics we will adopt the *interaction network* model:

Definition 28 (Interaction Network) *An interaction network $G = (V, E, T)$, is defined by a set of nodes V and a set of timestamped edges $E \subseteq V \times V \times T$ describing the interactions among them. An edge $e \in E$ is thus described by the triple (u, v, t) where $u, v \in V$ and $t \in T$. In particular, each edge e represents an interaction between nodes u and v that took place at a particular time t.*

To easily analyze an interaction network G we discretize it in τ consecutive snapshots of the same duration, thus obtaining a set of graphs $\mathcal{G} = (G^0 \dots, G^\tau)$ temporally ordered. When dealing with interaction networks, two different models can be used to discretize their evolution:

i) agglomerative growth, and

ii) interval bounded observations.

The former model assumes that, given two consecutive snapshot t and $t+1$ and their related networks $G^t = (V^t, E^t, T^t)$ and $G^{t+1} = (V^{t+1}, E^{t+1}, T^{t+1})$, all the nodes and interactions present in the first will be present in the second. More formally it assumes that holds:

$$\forall t, t+1 : V^t \subset V^{t+1}, E^t \subset E^{t+1}, T^t \subset T^{t+1}$$

Conversely, the latter splits the network following a memoryless assumption: the interactions belonging to G^t are only the ones that appear in the interval $(t-1, t)$. In the following we will adopt this last model since it allows us to make predictions not only for interactions that will take place among previously unconnected nodes but also to predict edges that have already appeared in the past. This decision is made in order to better simulate the dynamics that real interaction networks exhibit allowing nodes and edges both to rise and fall.

Due to the more complex model analyzed, hereafter we will refer to this peculiar formulation of the LP problem as the *Interaction Prediction* (IP in short) one. Given these preliminaries, we can now introduce its formal definition:

Definition 29 (Interaction Prediction) *Given an interaction network G, observed for each time $t \in T$, where $T = \{0, 1, \dots, \tau-1, \tau\}$, the* interaction prediction *problem aims to predict new interactions that will took place at time $\tau + 1$.*

In the following section we introduce our supervised approach to solve the interaction prediction problem.

[1] G. Rossetti, R. Guidotti, D. Pennacchioli, D. Pedreschi and F. Giannotti, "Time-Aware Interaction Prediction in Dynamic Social Networks exploiting Community Discovery", 2014

8.2.1 Time-Aware Interaction Prediction

The peculiar formulation of the interaction prediction problem introduces new challenges to an already complex task. Due to the evolutionary behavior of the networks subject of our investigation, a particular effort is needed in order to find a reasonable way to take care of structural dynamics during the prediction phase. Our idea is to make use of timestamped network observations and community knowledge besides classical features in order to learn a robust machine learning model able forecast new interactions.

We design our approach to follow four stages:

1. For each temporal snapshot $t \in T$ we compute a partition $\mathcal{C}^t = \{C_0^t, \ldots, C_k^t\}$ of G^t using a community discovery algorithm. Then we define, for each t and C, $G_C^t = (V_C^t, E_C^t)$ as the sub-graph induced on G^t by the nodes in C^t, such that $V_C^t \subseteq V_t$ and $E_C^t \subseteq E^t$.

2. We consider the interaction communities $\mathcal{C}^t$ of G and, for each $t \in T$, we compute a set of measures F for each pair of nodes pair $(u, v) \in W_{\mathcal{C}}^t$ such that $W_{\mathcal{C}}^t = \{(u, v) : u, v \in V_C^t \wedge distance(u, v) \leq \delta \wedge C^t \in \mathcal{C}^t\}$, that is (u, v) belong to the same community at time t and they are at most δ hops of distance in the shortest path connecting them. Thus we obtain values $f_{u,v}^t$ describing structural features, topological features and community features of the node pairs (u, v) at time t.

3. With these values, for each couple of nodes $(u, v) \in W_{\mathcal{C}}^t$ and feature $f \in F$ we build a time series $S_{u,v}^f$ using the sequence of measures $f_{u,v}^0, f_{u,v}^1, \ldots, f_{u,v}^\tau$. Then, we apply well-known forecasting techniques in order to obtain its future expected value $f_{u,v}^{\tau+1}$.

4. Finally, we use the set of expected values $f_{u,v}^{\tau+1}$ for each feature $f \in F$ to build a classifier that will be able to predict future intra-community interactions.

In the following we analyze each step by itself, proposing some instantiations of the proposed methodology able to address the IP problem.

Stage 1: Community Discovery

In order to evaluate the impact of community structure on the predictive power of the proposed methodology, we have decided to test three different CD algorithms, namely: Louvain, Infohiermap and Demon. In order to better highlight the characteristics of each algorithm, we report the their major peculiarity while in the experimental section we will discuss how they affect the predictive power of our methodology. We remind that we adopted community discovery algorithms to split interaction networks into communities, then we used these communities to both calculate the features that will be illustrated in the following and to perform the predictions of new interactions.

- Louvain, already introduced in 7.3, is a heuristic method based on modularity optimization [110]. It is fast and scalable on very large networks and reach high accuracy on ad hoc modular networks. The optimization is performed in two steps. First, the method looks for "small" communities by optimizing modularity locally. Second, it aggregates nodes belonging to the same community and builds a new network whose nodes are the communities. These steps are repeated iteratively until a maximum of modularity is attained and a hierarchy of communities is produced. Louvain produces a complete non-overlapping partitioning of the graph. As most of the approaches based on modularity optimization it suffers of a "scale" problem that cause the extraction of few big communities and a high number of very small ones.

- Infohiermap is one of the most accurate and best performing hierarchical non-overlapping clustering algorithm for community discovery [194] studied to optimize community conductance. The graph structure is explored with a number of random walks of a given length and

with a given probability of jumping into a random node. Intuitively the random walkers are trapped in a community and exit from it very rarely. Each walk is described as a sequence of steps inside a community followed by a jump. By using unique names for communities and reusing a short code for nodes inside the community, this description can be highly compressed, in the same way as re-using street names (nodes) inside different cities (communities). The renaming is done by assigning a Huffman coding to the nodes of the network. The best network partition will result in the shortest description for all the walks.

- DEMON is a fully incremental and limited time complexity algorithm for community discovery already introduced in 7.1.

We chose to use the aforementioned algorithms since, due to their formulations, they cover three different kind of community definitions: modularity, conductance and density based ones. Since in our test we vary the structural properties of the communities used to extract the classification features, in the experimental analysis we will be able to understand which network partitioning approach is able to provide better insights.

Stage 2: Features Design

Identify the right set of features to train a classifier is one of the most complex part of every supervised link prediction approach. We have decided to use information belonging to three different families: *pairwise structural features*, *global topological features* and *community features*.

a) Pairwise Structural Features

In this class fall all the measures used in literature to score the likelihood of new links in unsupervised scenarios. From hereafter, given a graph G, we will use the following notation: $\Gamma(u)$ identifies the set of neighbors of a node u in G; $|\bullet|$ represents the cardinality of the set $\bullet$.
Starting from the measures proposed in [63] we restricted our set to the following ones:

- *Common Neighbour (CN)*
  This metric computes assign as likelihood score of a new link the number of neighbors shared by endpoints [201]. More formally,

$$CN(u,v) = |\Gamma(u) \cap \Gamma(v)| \tag{8.15}$$

- *Jaccard Coefficient (JC)*
  This coefficient measures the likelihood of two nodes to establish a new connection as the ratio among their shared neighbors and the total number of distinct neighbors they have [202]. It is defined as:

$$JC(u,v) = \frac{|\Gamma(u) \cap \Gamma(v)|}{|\Gamma(u) \cup \Gamma(v)|} \tag{8.16}$$

- *Adamic Adar (AA)*
  This measure refines CN by increasing the importance of nodes which posses less connections [61]. Its formal definition is:

$$AA(u,v) = \sum_{w \in \Gamma(u) \cap \Gamma(v)} \frac{1}{\log |\Gamma(w)|} \tag{8.17}$$

- *Preferential Attachment (PA)*
  The PA measure assumes that the probability of a future link between two nodes is proportional to their degree [33]. Hence, it is defined as:

$$PA(u,v) = |\Gamma(u)| \times |\Gamma(v)| \tag{8.18}$$

As a direct consequence to their formulation CN, JC and AA share the same result set composed by all the pair of nodes at distance at most two in G: However, the values obtained by the three measures for the same edge do not correlate (i.e., having a high CN does not imply having high JC or AA). On the contrary, PA produces scores for all the possible pairs: to uniform its result set to the ones produced by other measures, in the following we consider this value only for the couples of node at most at distance $\delta = 3$ which belong to the same community.

b) Global Features

The features discussed so far look at the nodes immediate surroundings. However, also the position of a node within the network carries valuable information that can be exploited in order to predict which kind of nodes are attracted by it. In literature a wide set of measures were proposed to estimate the centrality of nodes and edges as well as their rank within a network. These scores are, often, computationally expensive to calculate: for this reason we have decided to make use only of two of them, specifically:

- *Degree Centrality (DC)*
  This measure is conceptually the simplest among the node centrality ones. It relates the centrality of a node with its degree: more formally,

$$DC(u) = |\Gamma(u)| \tag{8.19}$$

- *PageRank (PR)*
  PR is a link analysis algorithm introduced in [14] and used by the *Google* web search engine. It assigns a numerical score to each element of a hyperlinked set of documents with the purpose of measuring its relative importance within the set.

$$PR(u) = \frac{1-d}{N} + d \sum_{(u,v)\in E} \frac{PR(v)}{|\Gamma(v)|} \tag{8.20}$$

  where $PR(u)$ is the page rank score of node u, N is the total number of nodes and d is the damping factor. In our experimentation we used the default value for d (0.85).

The DC and PR scores were computed for both the endpoints of new edges: the underlying idea is to understand if there is some correlation among the centrality of two nodes and the likelihood of the appearance of a new interaction between them. This can be seen as a generalization of the PA measure where the operator defining the combination of the individual scores is not fixed.

Community Features

One of the most pressing issue related to link prediction regards the reduction of *false positive* forecasts. To this extent, as briefly mentioned before, we exploit community discovery as a way to reduce the number of predictions provided by the chosen pairwise structural features. Making predictions only between nodes belonging to the same community allows the predictive process to focus only on connections that are highly likely to appear, thus discarding the ones connecting different graph substructures. However, following the general intuition behind the idea of community, we can take advantage of more specifically designed measures: indeed, all the information we can gather from the topological analysis of the communities can be used as features describing the extended surroundings of nodes. With this aim we have decided to introduce the following features:

- *Community Size (CS)*
  The number of nodes belonging to the community C. Defined $G_C = (V_C, E_C)$ as the graph induced on G by the nodes in C, we have:

$$CS(G_C) = |V_C| \tag{8.21}$$

- *Community Edges (CE)*
  The number of edges within nodes in C, formally:

$$CE(G_C) = |E_C| \tag{8.22}$$

- *Shared Communities (SC)*
  Given two nodes $u, v \in V$ and a set of communities $\mathcal{C} = \{C_0 \dots C_n\}$ $CS(u, v, \mathcal{C})$ identifies the number of communities shared by u and v. When dealing with network partitions SC takes value in $\{0, 1\}$ while in case of overlapping communities its domain is $[0, |\mathcal{C}|]$.

- *Community Density (D)*
  Community density is computed as the ratio of edges belonging to the community over the number of possible edges among all the nodes within it.

$$D(C) = \frac{|E_C|}{|V_C| \times (|V_C| - 1)} \tag{8.23}$$

- *Transitivity (T)*
  T identifies the ratio of triangles with respect to open "triads" (two edges with a shared vertex) in G_C.

$$T = 3\frac{|triangles(G_C)|}{|triads(G_C)|} \tag{8.24}$$

- *Max Degree (MD)*
  MD identifies the degree (w.r.t. the community subgraph) of the principal hub for the community C.

$$MD(C) = max\{|\Gamma(u)| : u \in V_C\} \tag{8.25}$$

- *Average Degree (AD)*
  AD identifies the average degree (w.r.t. the community subgraph) of the nodes within the community C.

$$AD(C) = \frac{\sum_{u \in V_C} |\Gamma(u)|}{|V_C|} \tag{8.26}$$

Stage 3: Forecasting Models

The third stage of our approach involves the adoption of time series forecasting models in order to obtain, given subsequent observation of the same feature for the same pair of nodes, an estimation of its future value. Since the behavior of the observed time series is not known in advance, we adopt several forecasting models based on different assumptions. Since the time series we are analyzing are not large, we have decided to not employ complex models that are known to be very efficient on extended observation periods. In fact, we tested four simple and computationally efficient models that have shown to record good performances on short time series. In the following definitions we identify with $Z_t = (t = 1 \dots \tau)$ a time series with τ observations and with Θ_t its forecast at time t.

- *Last Value (Lv)*
  This first method considers as forecast the last observed value of the time series. The forecast is defined as:

$$\Theta_t = Z_{t-1} \tag{8.27}$$

- *Average (Av)*
  The forecast is given by the average of all the observations in Z_t:

$$\Theta_t = \frac{\sum_{i=1}^{T} Z_i}{\tau} \tag{8.28}$$

- *Moving Average (Ma)*
  This approach makes the prediction by taking the mean of the n most recent observed values of a series Z_t. In our experiments we have ranged n in the interval $[1, \tau]$.

$$\Theta_t = \frac{\sum_{i=\tau-n}^{\tau} Z_i}{n} \tag{8.29}$$

- *Linear Regression (LR)*
  Linear Regression fits the time series to a straight line. The level α and the trend β parameter (used to estimate the slope of the line) are defined by minimizing the sum of squared errors between the observed values of the series and the expected values estimated by the model. This forecast is defined as:

$$\Theta_{t+h} = \alpha_t + h\beta_t \tag{8.30}$$

Stage 4: Classifier Models

Predicting correctly new interactions is not an easy task. The complexity is mainly due to the highly unbalanced class distribution that characterizes the solution space: real world networks are generally sparse, thus the number of new links/interactions over the total possible ones tends to be small. We have seen how it is possible, at least to some extent, mitigate this problem by restricting the prediction set (i.e., by limiting the forecast to nodes at distance at most δ and predicting only new edges among nodes within the same community). However, even adopting such precautions we can expect a substantial unevenness between the positive and the negative classes. This translates into a very high, hard to improve, threshold for the baseline model (i.e., in case of a network having density=0.1, which identifies the presence of "only" $\frac{1}{10}$ of the possible edges, the majority classifier is capable to reach more than 0.9 of accuracy). To overcome this issue we adopted class balancing through downsampling (as performed in previous works, like [78]), thus obtaining balanced classes and a baseline model having 0.5 accuracy. Indeed we pursued such approach but, in order to provide an estimate of the real predictive power expressed by our methodology we also tested it against the real unbalanced class distribution.

Moreover, since the main focus of this work is to propose a mining approach to the IP problem and not to discuss a specific classification model, we have decided to evaluate our strategy independently from a specific classifier: for this reason in the following section we discuss results achieved by an ensemble of classifiers and we show the scores only for the best performing ones. In detail, we have tested six models: Decision Tree (C4.5, C&R, CHAID, QUEST), Neural Network and Logistic Regression.

Experiments and Results

Here we report the results obtained by applying our approach to two real world interaction networks. Firstly the datasets used to perform the experiments are briefly introduced, then the results obtained in a balanced class scenario are reported together with a deep analysis of the best performing community discovery algorithms, features and forecast models. Finally, we extend our experiments to evaluate the proposed approach on an unbalanced class scenario.

Datasets

We tested our approach on two networks: an interaction network obtained from a Facebook-like[2] *Social* network, and a co-authorship graph extracted from *DBLP*[3]. These datasets allow us to test our procedure on two different grounds: a "virtual" context, in which people share thoughts and opinions via a social media platform, and a "professional" one. The general statistics of the datasets are shown in Table 8.3, while a brief resume is the following:

[2] http://toreopsahl.com/datasets/
[3] http://dblp.org

Network	Nodes	Interactions	#Snapshots	k_{CC}	σ_{CC}	k_D	σ_D
DBLP	747,700	5,319,654	10 (years)	0.665	0.018	3.113e-05	9.602e-06
Social	1,899	113,145	6 (months)	0.105	0.015	8.600e-03	1.400e-03

Table 8.3: Statistics for the analyzed networks.

- *Social*: The Facebook-like social network originates from an online community for students at University of California, Irvine. The dataset includes the users that sent or received at least one message during 6 months. We discretize the network in 6 monthly snapshot and use the first 5 to compute the features needed to predict the edges present in the last one.

- *DBLP*: We extract author-author relationships (i.e., two researchers are connected iff they are co-author of at least one paper). The co-authorship relations fall in temporal window of 10 years (2001-2010). The network is discretized on yearly basis: we use the first 9 years to compute the features and set as target for the prediction the edges belonging to the last one.

To underlying the low density of the observed networks across the various snapshots, we report their average value (k_D) in Table 8.3 together with the average network clustering coefficient k_{CC} and their variances, σ_D and σ_{CC} respectively. We can immediately notice that the low variances guarantee the reliability of both the averages of density and clustering coefficient. For this reason, it is remarkable the fact that Social is more dense than DBLP even though its clustering coefficient is considerably lower than DBLP. This can be explained by the fact that, due to its nature, when a new interaction appears in DBLP, more than a couple of users is involved, creating automatically a complete clique, while, in Social, a new interaction just expresses the exchange of a direct message between the two users.

Balanced scenario

To better highlight how the proposed approach performs on real world networks, we need to compare the outcome of its instantiations varying the combination of community discovery algorithm and time series forecast models used. In the following analysis we will make use of some measures aimed at identifying the performances of the tested classifiers. In particular, we will use *accuracy* and AUC which are defined in terms of the confusion matrix of a binary classifier (see Table 8.4):

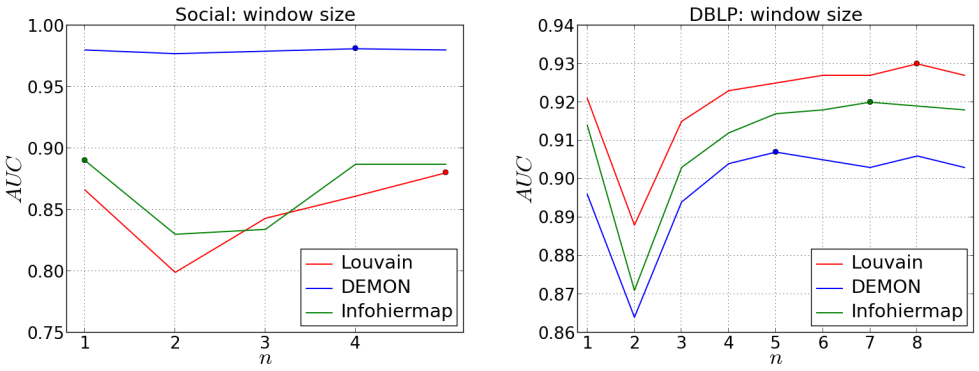
- *Accuracy*, defined as $ACC = \frac{TP+TN}{TP+FN+TN+FP}$, measures the ratio of correct prediction over the total;

- AUC identifies the area under the receiver operating characteristic (ROC). It is used to illustrate the performances of binary classifiers: it relates its True Positive rate ($TPR = \frac{TP}{TP+FN}$) with its False Positive rate ($FPR = \frac{FP}{FP+TN}$) providing a visual interpretation useful to compare different models on the same data.

We carried out a first preliminary study aimed at identifying, fixed the community discovery algorithm, the optimal window size n for the moving average (Ma) forecast. By definition the Lv and Av are special cases of the more general Ma: particularly, the former is equivalent to Ma when $n = 1$ while the latter when $n = \tau$. In Figure 8.5 are shown, for the three community discovery

		predicted	
		class 0	**class 1**
actual	**class 0**	TN (true neg.)	FP (false pos.)
	class 1	FN (false neg.)	TP (true pos.)

Table 8.4: Confusion matrix of a binary classifier.



(a) (b)

Figure 8.5: Moving Average (Ma): AUC values varying observation window size $n \in [0, \tau]$, dots highlight highest ones.

algorithms, how the classification accuracy behaves varying the observation window n.

We can observe different trends for Social and DBLP networks: in the former the AUC is maximized by the classifier built upon Demon communities, while in the latter the same approach is the one with lower performances. This is probably due to the particular definition of ego-network based overlapping communities provided by this approach which is tailored explicitly for social contexts. Furthermore, by observing these plots we can conclude that, in order to obtain higher performances using Ma, two strategies are consistent:

1. minimize n using as forecast the last value (Lv) in order to make inference approximating the future with the actual network status, or
2. use $n \simeq \tau$ in order to have a better estimation of the whole historical trends.

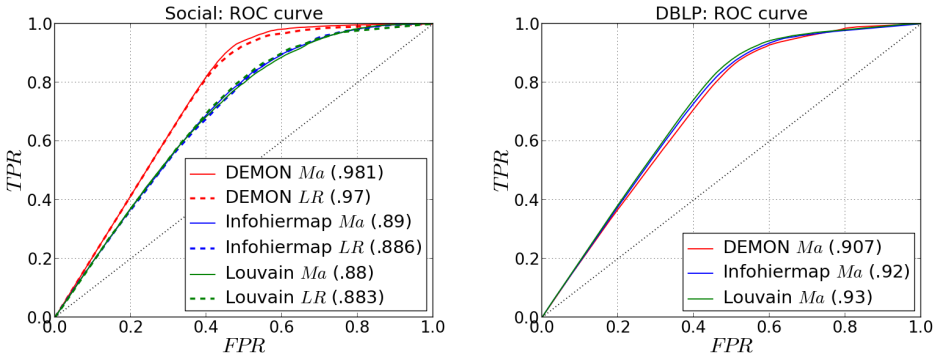
Hereafter, we will make use of the best scoring classifiers identified in Figure 8.5 to detail our analysis: we will refer to them as the Ma models for each specific network and community definition.

As second step we compare the outcomes of the classifiers built using the LR forecast models with the Ma ones. In Figure 8.6 are shown the ROC curves for both Social and DBLP datasets. In the former we can observe how LR and Ma provide very similar results even if the moving average is always capable to obtain slightly better performances. DBLP shows the same trend: a small gap between the two approaches (for this reason we omit the LR curve). Moreover, we report in Table 8.5 AUC and the ACC score for all the compared methods.

Once identified the two best performing classifiers for Social (Demon Ma and Infohiermap Ma) and for DBLP (Louvain Ma and Infohiermap Ma) w.r.t. AUC and ACC, we need to

Network	***DBLP***		***Social***	
Algorithm	**AUC**	**ACC**	**AUC**	**ACC**
Demon Ma	0.907	85.58%	**0.981**	93.55%
Demon LR	0.901	84.35%	0.970	91.87%
Louvain Ma	**0.930**	87.72%	0.880	80.27%
Louvain LR	0.926	87.48%	0.883	81.37%
Infohiermap Ma	0.920	86.69%	0.890	81.34%
Infohiermap LR	0.917	86.18%	0.886	80.89%

Table 8.5: Compared algorithms performances.



(a) (b)

Figure 8.6: ROC curves of the compared methods in balanced scenario for each community discovery algorithm. For Social *(a)* we draw LR and Ma forecast methods, while for DBLP *(b)* we draw only Ma ones.

investigate which are the key features that contribute to their performances. With this aim, we report in Figure 8.7 the relative importance of each feature used by the classifier for each method. We can see how in Social the classifier built upon DEMON *(a)*, as well as the one using INFOHIERMAP communities *(b)*, gives high importance to degree centrality and community measures (in particular density, size and average degree), and tends to make less discriminating decision using pairwise structural features (with the exception of PA). Conversely, in DBLP (bottom) the community features set seems to show small predictive power for both the analyzed algorithms. This discrepancy is probably due to the different nature of the studied networks: one, Social, naturally models real social interactions occurring in a short period, while the other, DBLP, is inferred from less volatile connections (working collaborations) that are developed through years.

In order to understand the boost provided to the classification task by the adoption of the right community discovery algorithm, we designed two different baselines: Structural Forecast (*SF*) and Filtered Structural Forecast (*FSF*). The SF model trains the classifier using only the forecasts for the pairwise structural features (CN, AA, PA and JC) computed on all the couple of nodes at distance at most 3 hops present in the whole network, not taking into account the presence/absence of shared communities among them. On the other hand, the FSF model restricts the computation to the node pairs belonging to the same community as the proposed approach does. As case study we report in Table 8.6 AUC and ACC of the best Ma and LR baselines for the Social dataset.

Since for this dataset our best performing approach is the one built upon DEMON communities, the structural features for the FSF baseline were computed using such network partition. The obtained results show that, using features extracted from the communities, we are able to gain 0.025 in AUC and 3.45% in ACC with respect to the FSF Ma baseline, and 0.08 in AUC and 10.67% in ACC with respect to the FS Ma one. This highlights the importance of communities for the interaction prediction task, not only in providing features for pair of nodes, but also in filtering the dataset in order to choose a more fitting selection of nodes for the prediction.

Algorithm	**AUC**	**ACC**
SF Ma	0.901	82.88%
SF LR	0.895	82.18%
FSF Ma	**0.956**	90.10%
FSF LR	0.937	88.09%

Table 8.6: Baselines for the balanced scenario on Social using Structural Forecast (SF) and Filtered Structural Forecast (FSF).

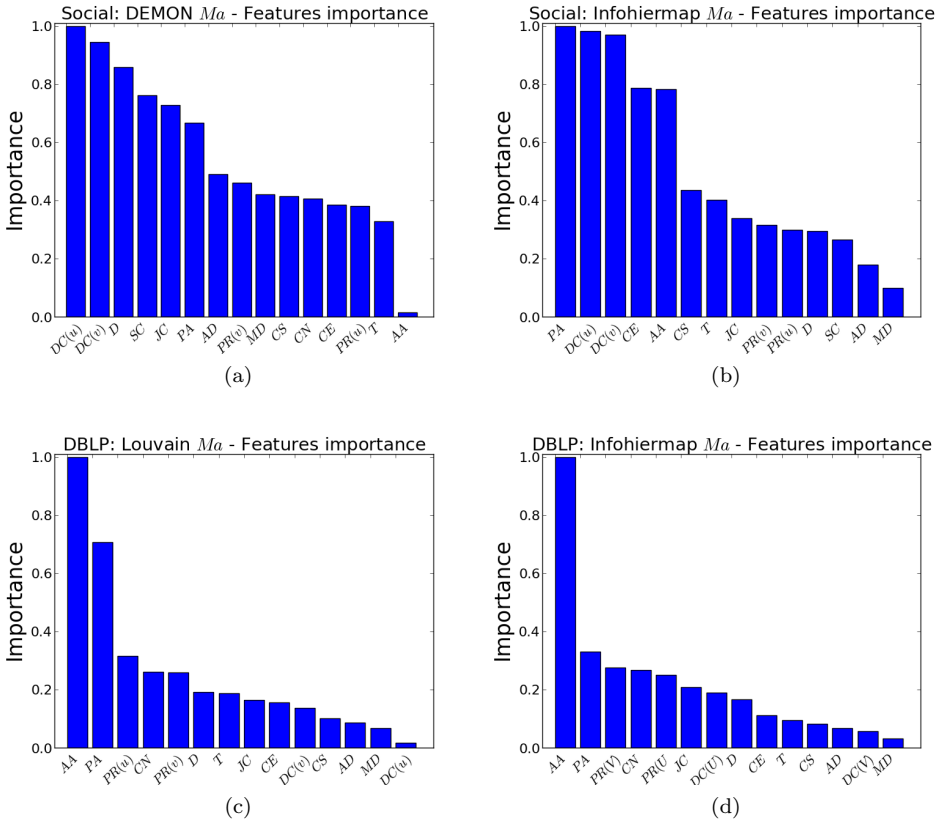


Figure 8.7: Features relative importance.

Without loss of generality, in the following of this section, in order to reduce the number of comparisons, we will report a full analysis only for the Social dataset. Furthermore, the results obtained for the DBLP scenario do not differ significantly from the ones discussed so far with the exception, as seen previously, of the best community discovery algorithm (Louvain instead of Demon). However, this divergence is due to the different nature and topology of the networks analyzed.

Feature Class Prevalence

Since our models are built upon three different classes of features (structural, topological and community related), it is mandatory to test their performances against the classifiers that use separately each one of them. Such analysis allow us to assess the predictive power of each class of features, giving an idea of their overall importance for the complete model. We built a classifier for each community discovery algorithm and each feature class by using together all the forecasted versions of the features belonging to it. As shown in Table 8.7, regardless the community discovery algorithm used, the most predictive features are the ones belonging to the *topology* class, followed by *structural* and *community* ones. However, we can observe how the AUC and ACC are always higher for the model based on the Demon approach: this trend suggests that such algorithm is the one that better bounds, at least for this network, the nodes that are more likely to establish future interactions.

Algorithm	**AUC**	**ACC**
DEMON *Structural*	0.957	90.59%
DEMON *Topology*	0.962	91.44%
DEMON *Community*	0.903	83.53%
LOUVAIN *Structural*	0.850	78.63%
LOUVAIN *Topology*	0.875	79.38%
LOUVAIN *Community*	0.724	66.64%
INFOHIERMAP *Structural*	0.876	79.85%
INFOHIERMAP *Topology*	0.887	80.81%
INFOHIERMAP *Community*	0.667	62.11%

Table 8.7: Compared classifier performances for different class of features on Social dataset.

Complete Classifier

In the following, we investigate if the good performances of the analyzed classifiers can be improved by combining all the features obtained at the end of the forecasting stage (i.e., all the time series forecasts computed with Ma and LR).

Algorithm	**AUC**	**ACC**
DEMON *All*	**0.981**	93.90%
LOUVAIN *All*	0.901	83.05%
INFOHIERMAP *All*	0.894	81.91%
FS *All*	0.959	90.44%

Table 8.8: Compared classifier performances using all the features obtained at the end of the forecasting stage on Social dataset.

As we can see in Table 8.8, the performance boost is negligible with respect to DEMON Ma, in fact we are able to gain only 0.35% in ACC maintaining the same AUC w.r.t. the results shown in Table 8.5. This means that the feature set used by our best classifier is "stable": its extension do not produce advantages that justify an increase of model complexity. Conversely, for LOUVAIN and INFOHIERMAP the gain in both AUC and ACC is more evident: this is probably caused by the different degree of approximation introduced for each feature in the forecasting stage.

Features Forecast Correlation

As consequence to the fact that there are only slight differences in performances when the classifiers are built using different forecasting methods, we decided to investigate which are the correlations among the forecasted values calculated by LR and Ma with $n \in [0, \tau]$. We analyzed each feature separately observing the correlation average, median, and variance. In Table 8.9 we report the

Algorithm	*Structural*	*Topology*	*Community*
DEMON	0.023	0.001	0.003
LOUVAIN	0.009	0.017	0.018
INFOHIERMAP	0.042	0.015	0.081

Table 8.9: Average of the variances of the correlations for different classes of features on Social.

average of the variances of these values aggregated for different class of features. From this Table emerges that, regarding structural features, LOUVAIN has the lowest average of variances of correlations, while, for topological and community related features, it is DEMON the approach which guarantee correlations. As result we can say that, if we use INFOHIERMAP (that has the highest average of the variances) to extract the communities from the interaction network, we should focus

on the choice of the different forecasting method applied. On the other hand, if we calculate the communities with DEMON, it does not matter very much which kind of forecast technique (LR or Ma) we use to calculate the expected values. This statement holds less strongly for LOUVAIN, that has a low correlation variance only for pairwise structural features.

Features Forecast Deviation

In order to fully understand the potential of the method proposed, we trained the classifier using the real values at time $\tau + 1$. This was done only with the aim of estimating the goodness of the expected value calculated with different forecasting method. Making the prediction on the Social dataset we observed very good performances close to those reached using the expected values, as we can see in Table 8.10.

Algorithm	**AUC**	**ACC**
DEMON	0.987	95.76%
LOUVAIN	0.888	81.16%
INFOHIERMAP	0.846	75.95%

Table 8.10: Compared algorithms performances with real values at time $\tau + 1$.

This is an indicator that a good approximation of the real values is important to build a reliable classifier. Thus, we analyzed the deviations of the expected values of the different forecasting methods with the real ones, calculating $((f_{u,v}^{\tau+1} - \hat{f}_{u,v}^{\tau+1})^2)$, and we detected the following interesting points.

With the exception of LOUVAIN, LR has the worse approximation with real values. This indicates that using DEMON or INFOHIERMAP it is better to apply Ma to forecast the expected values. Moreover, LOUVAIN is generally worse than the others for every feature, while INFOHIERMAP works better for structural and global topological and DEMON minimizes the error for the community ones (which is consistent with the results discussed during the analysis of the *complete classifier* scenario). The previous observations result from the analysis of the squared error distribution depicted in Figure 8.8. At any rate, independently from the community discovery algorithm or from the forecasting method, the deviation is nearly always very low justifying the good performances previously exposed.

As an additional analysis we also studied the Sum of Squared Error (SSE) for each forecasting method of each feature. By ranking the SSE among the various features (properly normalized in order to be comparable), we find out that, with respect to the other combinations, INFOHIERMAP with LR has the highest SSE for each attribute. On the other hand, the best approximations are achieved by INFOHIERMAP and DEMON with Ma with $n \in \{3, 4\}$. Indeed, with the exception of AA, LOUVAIN never has the lowest SSE among the features used. At the same time, by ranking the SSE among the different community discovery algorithms and forecasting techniques, it emerges that with LOUVAIN the lowest SSE belong to AA while the highest to SC. On the contrary, with DEMON the lowest SSE belong to SC, while the highest changes with respect to the forecasting method. Finally, as far as INFOHIERMAP concerns, we cannot derive nothing interesting. Thus, probably, due to its nature related to ego-networks, DEMON gives better results than the other community discovery algorithms for community features, while AA works really well with the communities extracted by LOUVAIN.

Unbalanced scenario

We have shown how our approach is able to obtain valid results in case of balanced class distribution. Unfortunately this scenario is not common when addressing the IP problem. Furthermore, making predictions on new interactions that will appear in a dynamic network involves, potentially, computing scores for all the $|V| \times (|V| - 1)$ pair of nodes of the network itself. Indeed, social tissues are very sparse: such topological connotation is the main reason for a high rate of False Positive

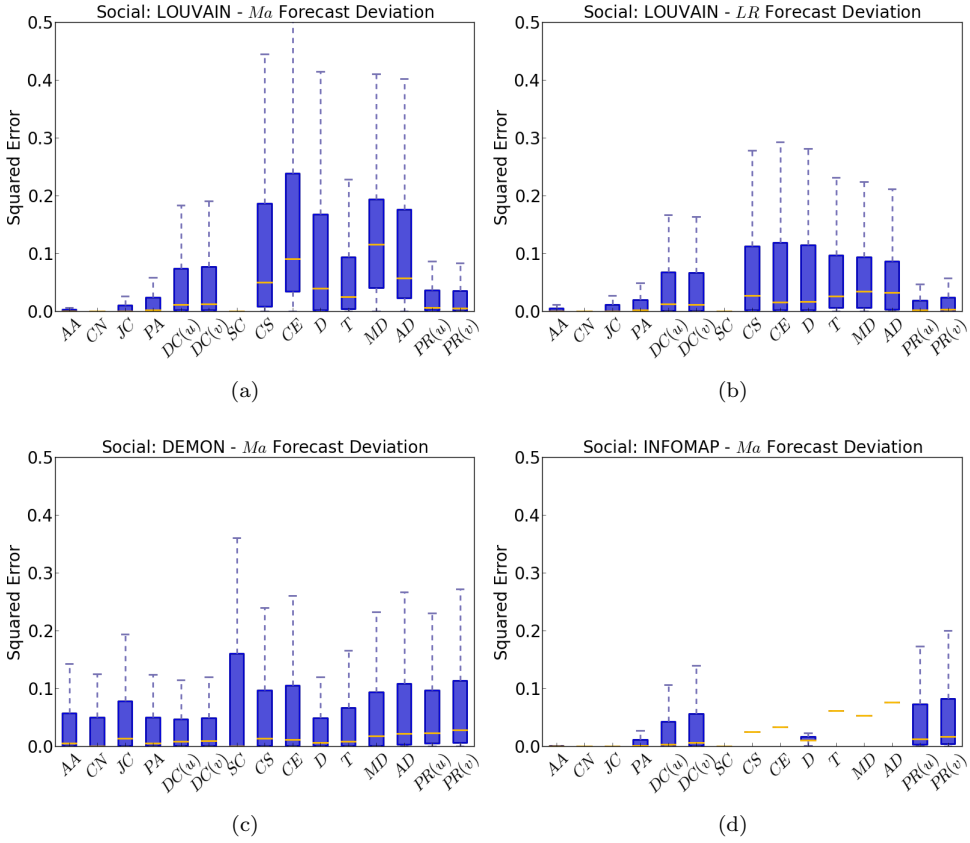


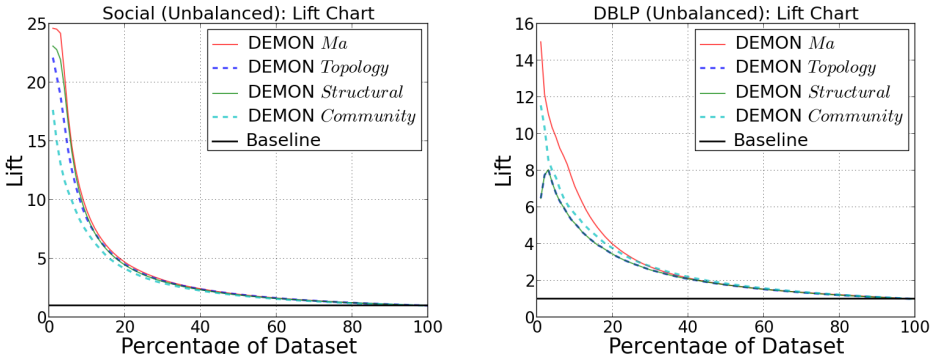
Figure 8.8: Boxplots of squared errors per feature. For lack of space reasons, we report just the most significant algorithms.

predictions as well as for naive overfitting models that maintain high accuracy just predicting the absence of new links (i.e., the majority classifier in case of supervised learning). Indeed, predicting every object as belonging to the larger class achieves a high performance in prediction, but it represents a useless classification strategy. For this reason, evaluate the performances of a classifier in highly unbalanced scenarios is not an easy, but very important, task. Since we want to predict correctly new links, our primary purpose is to reach high precision avoiding the generation of false positive predictions.

For this reason, as already done in 7.3, for the unbalanced scenario we will discuss, besides AUC and ACC, the *Lift Chart* and *precision* of the tested classifiers.

Here we report the precision instead of the accuracy because, unlike the balanced scenario (where starting from a ratio of $50-50$ the accuracy has a strong significance), in the unbalanced one it is very easy to get a high, but meaningless, accuracy. This is due to the fact that, as consequence to the sparsity of the interaction network, the majority classifier can predict always "no edge" with no effort reaching very high performances. Besides this we report the *Lift Chart* because, conversely from AUC and PPV (with which shares, describing isomorphic spaces, the conveyed information), it is able, even in unbalanced scenarios, to graphically emphasize the improvements provided by the tested classifier against a baseline model.

The datasets considered are the same ones analyzed in the balanced scenario, Social and DBLP: however, in this case we preserved the original ratio between the node pairs with and without a future interaction. We use for both networks the DEMON algorithm to extract communities. This choice is due to the following reasons:



(a) (b)

Figure 8.9: Lift Charts of the compared methods in unbalanced scenario.

- Social: DEMON reach the best performances in the balanced scenario, for this reason we expect that it will behave well even in unbalanced scenario;

- DBLP: Using LOUVAIN (i.e., the best performer in the balanced scenario) in the unbalanced scenario, all the classification models outputs the majority classifier.

In Social, the ratio of negative class over the total amount of possible pairs is 95.947%, that means that a majority classifiers predicting no edge for all the pairs would have an accuracy of almost 96%. As output from the classification phase with Ma we have a model which reaches an AUC of 0.966 with a prediction accuracy of 98.75% and a precision w.r.t. the positive class of 95.61%. These are very significant results: on one hand we have an accuracy improvement of 2.803% in an ideal window of 4.053% (100% − 95.947%) with respect to the majority classifier while, on the other, we have a very high precision on the positive class, considering that a classifier predicting always an edge would have a precision of 4.053%.

In addiction to the Ma model, we also build three classifiers each one of them considering all the forecasts for a single category of features: topological, structural, and community.

In Figure 8.9*(a)* we show the Lift Chart of the four models applied to Social. From the chart emerges that after the Ma model, the most promising is the one built upon the topological features followed by structural and community ones.

Also in this highly unbalanced scenario, we want to "measure" how much the chosen community approach impacts the efficiency of our workflow filtering in the "promising pairs". By building the dataset with all possible pairs without the community discovery approach, we have the majority class, i.e., the absence of link, with a ratio of 98.96% over the total number of entries. In order to better compare the two cases, we filter out randomly some pair with no edge, bringing the accuracy of the majority classifier at 95.947% (like in the case with community discovery). Again we compare the performances for the SF and FSF, reported in Table 8.11, but now considering the precision instead of the accuracy. We can see that we gain almost a 10% of precision just filtering out, in any time slot, all the pairs not belonging to the same community.

Algorithm	**AUC**	**PREC**
SF Ma	0.897	64.06%
SF LR	0.893	62.62%
FSF Ma	0,918	**74.71**%
FSF LR	**0.932**	72.45%

Table 8.11: Baselines for the unbalanced scenario on Social using SF and FSF.

In DBLP case study, the resulting classifier has an AUC of 0.86, an ACC of 98.135% and a precision with respect to the positive class of 44.78%. The majority class (no link) has a ratio of 98.13% over all the instances of the dataset. A possible reason of the lower performances obtained on DBLP w.r.t. Social ones, is that in the latter an interaction represent a real social action between two different actors, while in DBLP an interaction model a relation of co-authorship in a paper, and the co-authorship is not, in our opinion, a strong representative of social interaction (see 5.1.2). However, we can notice that the performances are not completely bad: we have a precision of 44.78%, starting from a ratio of positive class of 1.865% (100% − 98.135%), that is 24 times better than predicting for any pair the presence of the edge. Finally, we can observe from the Lift Chart in Figure 8.9*(b)* how, differently from the Social case, the most predictive set of features are the community ones, over the structural and topological.

Discussion

In this work we have tackled the Interaction Prediction problem in a dynamic network scenario. Since networks often model rapidly evolving realities that cannot easily be *"frozen"* in time without loss of information, a time-aware approach to link prediction is mandatory to achieve valuable results. Moreover, due to the intrinsic high computational cost of the approaches that solve this problem, it is important to reduce the list of possible candidates for which to compute a prediction (preferably avoiding the generation of false positives). To this extent we have exploited the community structure of social networks to both bound the result set, and design features whose analysis through time allows the description of a high performance supervised learning strategy. The results obtained with the proposed methodology open the way to several future lines of analysis. Indeed, more accurate time series forecast techniques can be evaluated in order to reduce the forecast error and evolutionary community discovery approaches can be used in order to incorporate community life-cycle features within the predictive process. Moreover, w.r.t. the type of dataset used, it could be possible to consider other classes of features such as mobility knowledge and spatial co-location. All these improvement will lead to more narrow and sophisticated classifiers that, taking into account an more heterogeneous feature set, will be able to better predict future human interactions.

Chapter 9

Modeling Collective Dynamics

> Time changes everything except
> something within us which is always
> surprised by change.
>
> — *Thomas Hardy*

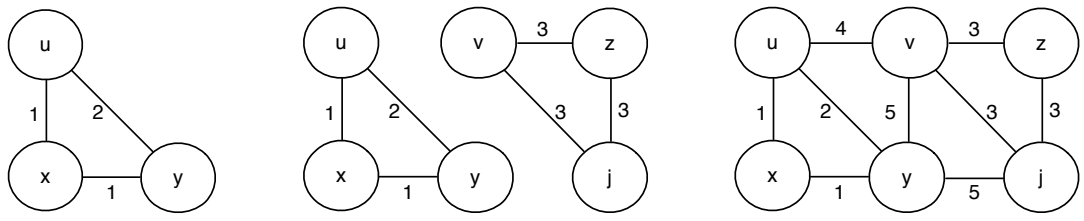
In the 2^{nd} part of this thesis, we have already seen that Community Discovery is one of the most discussed problem within the complex network analysis field. There, we have already proposed a valid approach (Demon, 7.1) that has shown to be a good fit for the analysis of social contexts: indeed, we have shown how, in such settings, the structures it defines are able to bound homophily and how this information can be used in order to address different tasks (namely, Quantification 7.2 and Social Engagement 7.3). However, even if the problems we have studied so far with the support of our algorithm are shaped by an intrinsic dynamism, Demon does not provide any support to directly exploit such information.
As observed in 8.2, Social networks often describe highly dynamic realities: in these cases we are prone to consider them as composed by volatile *"interactions"* more than permanent *"ties"*. In these settings the community discovery problem needs to be reformulated: nodes changes their connectivity as time goes by and this affect their involvement into communities. For this reason in 9.1 we introduce a novel algorithm, Tiles, which builds and maintains updated communities by continuously looking at the interactions that occur among the nodes of the network. This approach will enable us to an online observation of communities and to a more fine grained analysis of their life-cycles.

9.1 Evolutionary Community Discovery[1]

As we have extensively discussed in 7.1, the concept of "community" in a complex network intuitively depicts a set of individuals that are very similar, or close to each other more than to anybody else outside the community. However, a completely shared formal definition of "community" does not exist: different approaches capture different characteristics and are based on a wide variety of assumptions. Given those circumstances, it is very difficult assess the goodness of the communities identified by a specific algorithm: for this reason each family of approaches proposes its own evaluation metric tailored to capture the properties modeled by the extracted structures. Until recent years, community discovery algorithms were proposed to deal only with static graphs. This choice underlines a QSSA (quasi-steady state approximation) assumption: networks can be frozen in time because mutation in their topology happens only in the long run. Indeed, dealing with flattened networks simplifies the formulation of algorithms aimed to extract knowledge from them: however this assumption cannot be always satisfied.

Indeed, networks are often used to model complex and rapid-scale human dynamics: social interactions, call graphs, buyer-seller scenarios are all examples of realities for which a QSSA assumption is extremely stringent and, inevitably, lead to analytical results that overestimate (or underestimate) the real connectivity. The aforementioned interaction networks reveal scenarios in which a static community discovery approach fails to identify meaningful knowledge. A community extracted from a social network, without taking into account the temporal ordering and the delay of interactions, will group together agents that have been in contact rarely and whose interactions can be very distant in time one from the others (as shown in Figure 9.1). These structures, that doubtless satisfy the community quality function for the adopted algorithm, do not reveal a picture consistent with the actual reality. To overcome this limitation the classical community discovery problem has to be reformulated: as done for clustering approaches [126] we need to introduce an evolutionary variant able to deal with rapidly evolving scenarios. Several strategies have been proposed to identify and track communities and their lifecycle. However, only few of the algorithms proposed so far approach in a direct way the following problem: evolutionary interaction-based community detection on highly dynamic networks.
In this work, we propose a solution for the aforementioned problem presenting Tiles (a.k.a. *"Temporal Interactions: a Local Edge Strategy"*), an evolutionary community discovery algorithm which follows a domino effect. In order to apply Tiles, the observation period does not need to be split in fixed temporal snapshots: each time a new interaction took place the community memberships for the new edge's endpoints and adjacent nodes are re-evaluated. Using a call graph of one million users, a Facebook interaction networks, an interaction network of 8 million users of a Chinese

[1] G. Rossetti, L. Pappalardo, D. Pedreschi and F. Giannotti, "Tiles: Evolutionary Community Discovery", 2014



(a) t=2 (b) t=3 (c) t=5

Figure 9.1: Communities identified by a dynamic community discovery algorithm. Numbers on the edges represent the interaction time. A static algorithm does not take into account the network evolution and would only identify one community (c). An evolutionary algorithm conversely finds several different communities since the network is observed during different stages of its evolution: if observed at $t = 2$ (a) we have the community $C_0 = \{u, x, y\}$; at $t = 3$ a new community $C_1 = \{v, z, j\}$ appears and for $t = 5$ all nodes are part of a single community.

microblogging platform observed for one year and synthetic benchmark graphs with ground truth communities we will show a comparison between TILES results and those produced by classical CD algorithm, highlighting how a direct interaction-based approach reveals new and interesting community patterns.

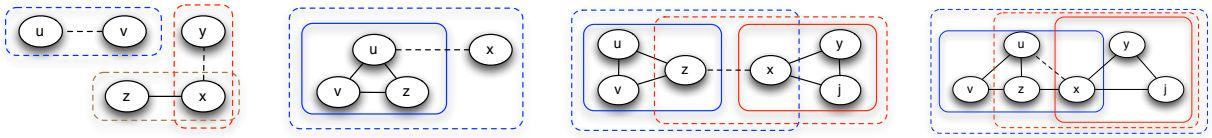
Evolutionary Community Discovery

The overwhelming number of papers on Community Discovery proposed in recent years express clearly the real issue to address: researchers are not interested in formulating *"The Community Discovery algorithm"* but in finding the right algorithm for each specific declination of the problem. Following this path, our approach aims to cope with a specific and not yet deeply studied scenario: community discovery in dynamic interaction networks.
Networks are dynamic objects: online social interaction networks, call graphs, economic transactions are all sources of information that evolve over time. As we have extensively discussed, the rise of new nodes and edges can lead to deep mutations of network topology. An analysis that considers dynamic networks as static entities - frozen in time - necessarily introduces bias on its results. For this reason, while modeling dynamic phenomena the Community Discovery problem needs to be revised and its formulation extended:

Definition 30 (Evolutionary Community Discovery) *Given an interaction streaming source S and a graph $G = (V, E)$, where V is the set of nodes and E the set of timestamped edges ($e \in E$ is defined as a triple (u, v, t) where $u, v \in V$ and $t \in \mathbb{N}$ is the time of the edge generation by S) the Evolutionary Community Discovery problem (henceforth ECD) aims to identify, and maintain updated, the community structure that compose G as new interactions are generated by S.*

The interaction streaming source S models the rising of new interactions among pair of nodes without any restrictions: particularly, the interaction endpoints can be already part of the graph or newcomers which join the network for the first time. The ECD problem models scenarios where interactions among entities do not occur with a rigid temporal discretization but, conversely, flow "as a stream" as time goes by. After all, this is how our social interactions actually take place: phone calls, SMS messages, tweets, Facebook posts do not appear at predetermined time slots, but they are produced in a fluid streaming fashion. Consequently, the social communities we are involved with change their topology fluidly over time. For this reasons, a valid algorithm for the ECD problem needs to be able to answer the following question: given a community C at timestamp t and a streaming source S, what will be its structure at an arbitrary time $t + \Delta$? In contrast with static community detection algorithms, an algorithm designed for the ECD problem needs to produce a series of observations of communities through time.

ECD models all those situations in which several interactions could occur among each couple of entities during the observation period. The community extraction can be approached following two different assumptions: (i) networks evolve in an accumulative fashion (i.e., once appeared edges and nodes are stable and will not disappear) or (ii) networks evolve gradually changing some entities from an observation to a subsequent one (i.e., edges and nodes can leave the network after a certain period). The first assumption fits the problem when a very strong relationship among nodes is defined for the network. We can apply such idea, for instance, to identify communities on a graph composed by several connected family trees: in this case the relationships expressed by edges ("be relatives") is a very strong one (even if seen rarely, a distant uncle remains still a relative, as well as the parent we live with). However, such example models a very specific kind of networks for which edges and nodes appear and disappear rarely. Real social networks are often built on less strong relationships definitions (as we have seen in 6.2) where edges establish instantaneous connections that in time decrease their strength until becoming useless. For instance, in a call graph we can imagine that the social strength of a phone call weakens after a certain time: the same thing happens for users' interactions on OSNs as Facebook or Twitter. In these general cases the second assumption seems to be the most meaningful one. One problem of the "limited



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

Figure 9.2: The appearance of a new edge: dashed line edges highlight new interactions; dashed line rectangles identify the periphery of communities; continuous line rectangles identify their core. (a) both nodes appears for the first time or one of them already exists but is not central; (b) a new node joins a strong community and become part of its periphery; (c) a new interaction occurs between nodes which are central in different communities: the endpoints become part of the periphery of the new joined communities; (d) a node, central for a community and peripheral in a different one, becomes central in the latter.

memory" growth assumption is how to find an acceptable temporal threshold that could be used to describe the TTL (time to live) of interactions. Such problem is context dependent: due to the different semantics of networks interactions, all the choices that can be made to overcome this issue are necessarily heuristics.
In the following section we introduce an algorithm capable to satisfy both the proposed scenarios relying on a single parameter: the expected TTL for the edges.

9.1.1 The Tiles algorithm

Social interactions determine how communities form and evolve. Even the appearance of a single new edge in the network leads to perturbations of the communities equilibrium. A common approach in literature to address CD in dynamic contexts is to split the network into temporal snapshots and repeat a static community detection for each snapshot, in order to study the variation of its mesoscale structures as time goes by. This approach, however, introduces an evident issue: which temporal threshold has to be chosen to partition the network? This problem, which is obviously context dependent, also introduces another issue: once the algorithm is performed on each snapshot how can we identify the same community in consecutive time slots?

To overcome those issues we have designed Tiles, an algorithm that does not impose fixed temporal thresholds for the partition of the network and the extraction of communities. It proceeds in a streaming fashion updating the observed communities whenever a new interaction is generated by the streaming source. As a fall of a domino tile, every time a new interaction emerges in the network, Tiles exploits a label propagation procedure to propagate the changes to the node surroundings, adjusting the neighbors' community memberships. According to Tiles, the belonging of a node to a community can be of two types:

- *weak* membership, which identify nodes in the "periphery" of the community (peripheral nodes);
- *strong* membership, for nodes in the "core" of the community (core nodes).

If a node is involved in at least a triangle with the others belonging to the same community it is defined as a core node, otherwise a peripheral one. Only core nodes are allowed during the label propagation phase to spread community membership to their neighbors (which become peripheral nodes if they do not participate in any triangle within the core). Tiles is an overlapping algorithm, i.e., each node can belong to different communities, which can represent the different spheres of the social world of an individual (friends, work, etc.).

The algorithm takes as input four parameters: (i) the graph G, which is initially empty; (ii) an edge streaming source S; (iii) τ, a temporal observation threshold; (iv) a Time To Leave value for

Algorithm 7 TILES($\mathcal{G}$, S, τ, ttl)

Require: $\mathcal{G}$: undirected graph, S: streaming source, τ: temporal observation threshold, ttl: time to leave

```
1: actual_t = 0
2: while S.isActive( ) do
3:     e←S.getNewInteraction( )
4:     G.removeExpiredEdges(ttl, actual_t)
5:     if e ∉ G then
6:         G.addEdge(e)
7:     end if
8:     if |Γ(e_u)|==1 & |Γ(e_v)| > 1 then
9:         WeakPropagation(e_u, e_v)
10:    else if |Γ(e_u)| > 1 & |Γ(e_v)|==1 then
11:        WeakPropagation(e_v, e_u)
12:    else
13:        CN ←Γ(e_u) ∩ Γ(e_v)
14:        if |CN| == 0 then
15:            WeakPropagation(e_u, e_v)
16:            WeakPropagation(e_v, e_u)
17:        else
18:            StrongPropagation(e_u, e_v, CN)
19:        end if
20:    end if
21:    if e_t - actual_t == τ then
22:        OutputCommunities(G)
23:        actual_t = e_t
24:    end if
25: end while
```

the interactions. The temporal observation threshold τ specifies how often we want to observe the structure of the communities, and allows us to customize the output of the algorithm. Furthermore, the TTL impacts the overall stability of the observed phenomena. Studying a dynamic network we can model its evolutionary behavior to comply to one of the following general scenarios: (a) *accumulative* or (b) *limited memory* growth. The former assumes that once an interaction among a pair of nodes has taken place it has to be considered permanent; conversely, the latter states that interactions gradually lose their strength as time goes by till disappear. TTL will be used to interpolate this two behaviors.

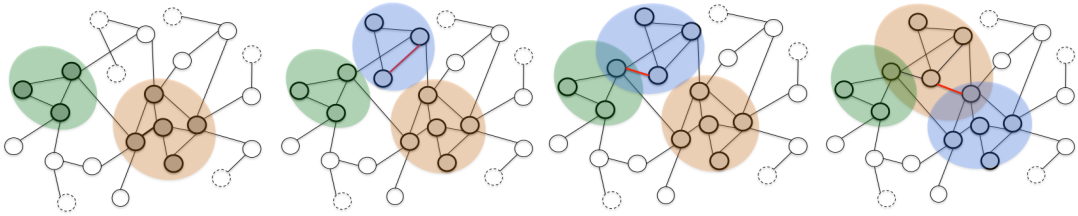
The execution of the algorithm produces as output, for each node, a series of timestamped observations: one every τ. Each observation is composed by two sets: the weak community memberships and the strong community memberships of the node. Setting τ equals to the streaming source clock ensures a punctual observation of community status on each new network update. The behavior of TILES is shown in Algorithm 7.

First of all (line 3-7) the new edge $e = (u, v)$ generated by the source S is added to the graph. Then the following scenarios are considered:

1. both the nodes u and v appear for the first time in the graph. No other actions are performed until the next interaction is produced by the source (Figure 9.2(a));

2. one node appears for the first time and the other is already existing but peripheral. Since peripheral nodes are not allowed to propagate the community membership, no action is performed until the next edge is produced by the source (Figure 9.2(a)). The same case applies when both nodes are existing but peripheral;

3. one node appears for the first time in G, while the other is an already existing core node (line 8-11). The new node inherits a weak community membership from the existing core node (Figure 9.2(b));



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

Figure 9.3: Community Growth: four consecutive updates extracted from a Facebook interaction network (FB07 network). Colors identify "core" communities. "Periphereal" nodes are identified by solid lines while nodes that are not involved in communities by dashed lines. New interactions are shown in red.

4. both nodes are core ones already existing in G (line 13-18). In this case two sub-scenarios can emerge:

   (a) Nodes u and v do not have common neighbors (line 14-16): they propagate each other a weak community membership through the WeakPropagation procedure (Figure 9.2, bottom left).

   (b) Nodes u and v do have common neighbors (line 18): their community memberships are re-evaluated and the changes propagated to their surroundings by the StrongPropagation function (Figure 9.2(c-d)).

Tiles communities grow gradually expanding their core and their peripheries through the WeakPropagation and the StrongPropagation procedures. The WeakPropagation procedure regulates the events in which a new node becomes part of an already established community. Since the newcomer is not involved in any triangle with other nodes of the community it becomes part of its periphery. The same function is performed when a new interaction connects existing nodes that does not share any neighbors. The StrongPropagation procedure assumes that the nodes u and v have at least a common neighbor z. For each triple (u, v, z) if at least two nodes are core for the same community the third one becomes core as well (example in Figure 9.3(c-d)), otherwise a new community is created upon the new triangle (example in Figure 9.3(a, b)). Once the core nodes are established, they propagate a weak membership to their neighbors, if they are not already within the community.

Due to its streaming definition, the complexity of Tiles is mainly shaped by the edge insertion phase, which can cause perturbation on the network topology and induce updates on the community structure. In the worst case scenario, this step has complexity $O(|V|)$ since the most costly rule is applied when the edge endpoints u and v share $|V|$ neighbors. However, reaching such upper bound is unusual due to the power law degree distribution which characterizes real world interaction networks: in such specific scenario the probability of having a node with degree k is $\backsim k^{\gamma}$ (where usually $2 \geq \gamma \geq 3$).

Expired edges removal

Tiles provides a valid solution to the ECD problem for the accumulative network growth scenario. If we want to allow nodes and edges to leave the network we need to introduce a removeExpiredEdges procedure (Algorithm 8) that follows a strategy consistent with the proposed community definition (line 4 of Algorithm 7). Tiles execution is parametric on a time to live value for interactions: when *ttl* is set to 0 an edge disappears immediately after its rising (causing an empty network at each new step); when *ttl* is set to $+\infty$ we fall in the accumulative growth scenario described so far. When *ttl* assumes value in $(0, +\infty)$ each edge ceases to exist after a *ttl* time from its generation by the streaming source S.

Once expired, an edge is removed from the graph (Algorithm 8, line 2). To improve the readability of the code, we report a linear search for expired interactions among all the ones present in the graph: however, the actual implementation uses a priority queue in order to minimize the number of tested interactions. The removal of an edge (u, v) causes the re-evaluation of community memberships both for nodes u, v and some other community members (i.e., their first level neighbors, lines 3-5). If after the removal of the expired edge the original community is broken into multiple connected components (lines 11-15), each one of them is considered as a new community. In order to assign each node to the *periphery* or the *core* of a community, the UpdateNodeRoles procedure (Algorithm 9) is called. This function analyzes the local clustering coefficient (CC) of each node within the specific community and retain as *cores* the ones with $CC > 0$ (Algorithm 9, lines 3-5) and as *periphereal* ones with $CC = 0$ (lines 6-11). Once ensured that the node roles within the modified community are consistent, a propagation is performed on the neighborhood of nodes which moved from the *core* to the *periphery*. Figure 9.4 shows an example of edge removal scenario.
The edge removal phase has cost $O(|RQ_{ttl}| * |to\_update|^3)$, where $O(|to\_update|^3)$ is due to the clustering coefficient computation (a naive implementation has cubic cost on the number *to_update* of nodes whose role have to be updated) and $RQ_{ttl} \subset RQ$[2] is the set of interactions candidate for the removal for the specific *ttl*. As $ttl \to 0$, $|RQ_{ttl}|$ becomes small while when $ttl \to \infty$ it increases its size: in the latter case the removal phase were executed rarely (until were not executed at all when *ttl* exceeds the observation period available for the data). For those reasons, this value can be seen as a constant.

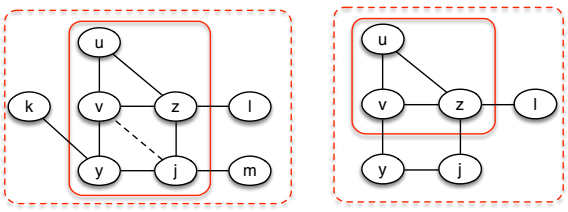
Tiles properties

Due to its streaming nature, Tiles shows two main properties: (i) It can be used incrementally on a precomputed community set; (ii) it can be parallelized if specific conditions are satisfied. Moreover, in presence of a deterministic interaction source S, Tiles output is uniquely determined. In the following section, we discuss and formalize such characteristics.

Property 5 (Incrementality.) *As specified above,* Tiles *is called on an initially empty graph. However, it also works when a non-empty graph and a set of precomputed communities are passed as parameters. Given a deterministic streaming source S_t at time t and a non-empty graph G_t, whose nodes are assigned to a community set $C_t = (c_1, c_2, \ldots, c_n)$,* Tiles *has the incrementality property:*

$$TILES(G_t, S_t) = TILES(G, S_0) \tag{9.1}$$

[2]RQ stands for Removal Queue.



(a) (b)

Figure 9.4: Expired edge removal. *(a)* the actual community where the edge (v, j) is candidate for removal; *(b)* the updated community: node j and y leave the community core because not involved in a triangle with central nodes (clustering coefficient equals to 0), node k and m leave the community periphery due to the propagation phase.

Algorithm 8 removeExpiredEdges(ttl,$actual_t$)

Require: ttl: edges time to live, $actual_t$: actual timestamp.
for all e in E **do**
  if ($actual_t$-e_t)$\leq$ ttl **then**
    removeEdge(e)
    shared_communities = communities shared by e_u and e_v
    to_update = $\{\Gamma(e_u) \cap \Gamma(e_v)\} \cup \{e_u, e_v\}$
    for c $\in$ shared_communities **do**
      components = getComponents(c)
      if |components| == 1 **then**
        UpdateNodeRoles(c, to_update)
      else
        for subcom $\in$ |components| **do**
          sc = NewCommunity(subcom)
          RemoveNodes(c, subcom_nodes)
          UpdateNodeRoles(sc, subcom)
        end for
      end if
    end for
  end if
end for

Algorithm 9 UpdateNodeRoles(c, to_update)

Require: c: community, to_update: set of nodes.
$\mathcal{C}$ = Subgraph(c)
for node $\in$ to_update **do**
  if ClusteringCoefficient(node)>0 **then**
    $c_{central} = c_{central} \cup \{node\}$
  else
    if c $\in$ $n_{central}$ **then**
      $n_{central} = n_{central} - \{node\}$
      $n_{periphery} = n_{periphery} \cup \{node\}$
      weakPropagation(node, c)
    end if
  end if
end for

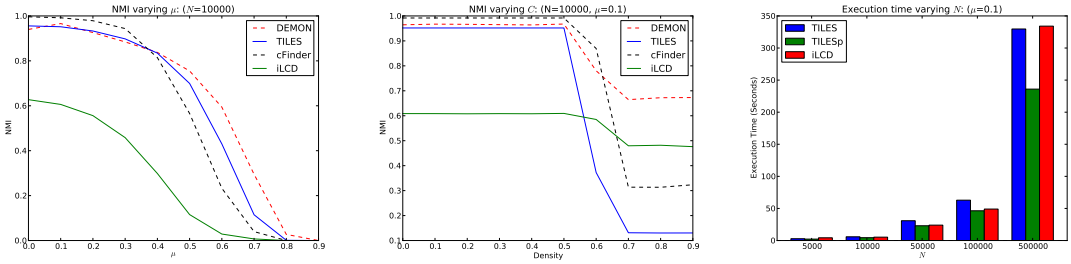
where G is an empty graph, C is an empty community set, S_0 is the streaming source S at the initial time. Since Tiles *is incremental, the final community evolutions produced starting with the source S at time 0 or at time t are identical, assuming that the streaming source is deterministic.*

This property provides the foundations on which Tiles is built: it describes the incremental nature of the algorithm update process. Given a Tiles-valid community set at time t this property ensure that, applying Tiles, the final result will be valid as well. Indeed, using as starting point a community partition which do not satisfy Tiles constraints it is not possible to make any assumptions on the final adherence of the identified substructures with the ones that would have been find by the same approach starting from scratch.

Property 6 (Compositionality.) Tiles *is parallelizable by identifying disjoint streams of edges produced by the deterministic streaming source S. Given a graph G, and two disjoint stream of edges S^i, S^{ii} iff $\forall (u_1, v_1) \in S^i$ $(u_2, v_2) \in S^{ii}$: $(c(u_1) \cup c(v_1)) \cap (c(u_2) \cup c(v_2)) = \emptyset$, where $c(\cdot)$ returns the set of communities the node is part of, then:*

$$TILES(G, S) = TILES(G, S^i) \cup TILES(G, S^{ii}) \tag{9.2}$$

The underlying idea is to operate updates on network subgraphs that are disjoint w.r.t. the communities assigned to the nodes: this is made possible by the constrained label propagation used



(a) (b) (c)

Figure 9.5: Comparison of several community detection algorithms on the same case. *(a)* NMI vs. μ; *(b)* NMI vs. network density; *(c)* runtime vs. network size.

to spread community membership. Moreover, this property also holds for the edge removal phase (considering the list of edges to be removed at each iteration as a streaming source). Isolating interactions among nodes of different communities makes possible to parallelize the algorithm, allowing a speed up of the computation of community evolution. We will briefly discuss the impact this property has on the runtime when evaluating our algorithm on synthetic data.

Experimental Results

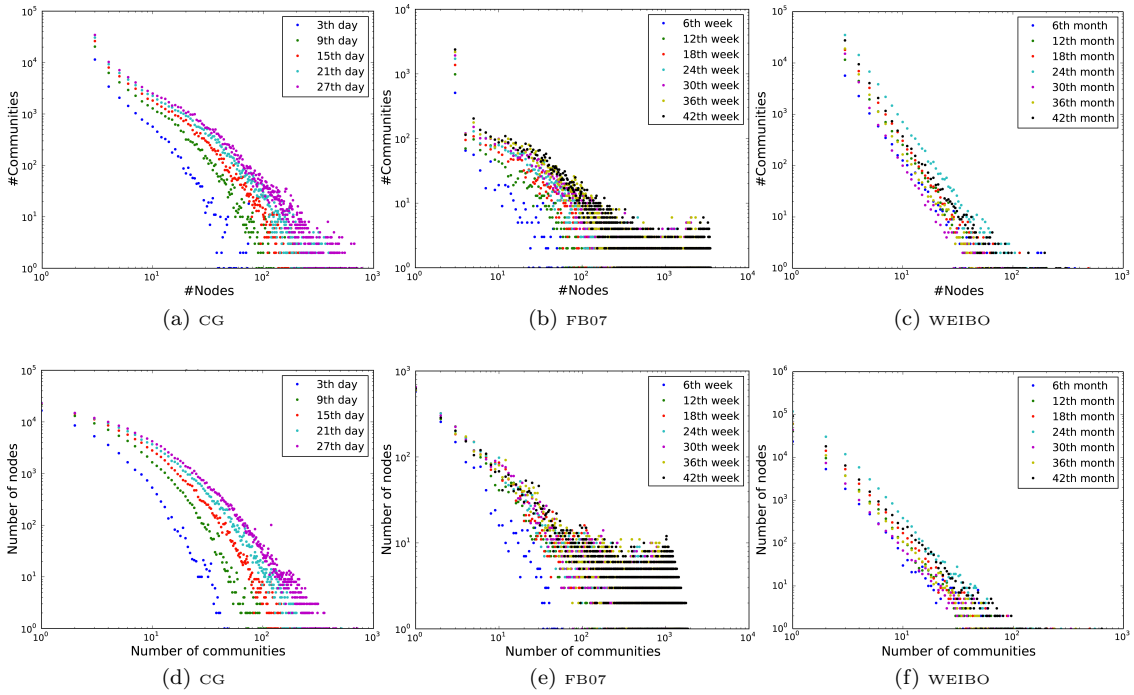
Evaluating results provided by a community discovery algorithm is a complex task, since a shared and universally accepted definition of what a community is does not exist. In literature, each algorithm proposes its own idea of a community and of the properties nodes should share in order to belong at the same partition of the network. Moreover, different Community Discovery algorithms are often designed to solve slightly different problems (i.e., overlapping and non-overlapping communities, static and dynamic communities, modularity-based and density-based communities). In the following we propose a validation of Tiles against other algorithms on synthetic networks with ground truth communities. Then we characterize the communities produced by Tiles on two real datasets of social interactions. Finally, we discuss the impact of the TTL parameter on community lifecycle when using Tiles in an edge removal scenario.

Evaluation on Synthetic networks

A comparison with the communities produced by different algorithms is the most straightforward way to assess the strengths and weakness of Tiles. Hence, in this section, we compare our algorithm with other static and dynamic community detection ones, using for Tiles an accumulative growth scenario ($ttl = \infty$, no edge removal). The plethora of community definitions introduced by different approaches makes questionable a direct comparison of the outputs obtained by two algorithms on the same network when a ground truth is not provided. Unfortunately, datasets with ground truth, i.e., real partition of the network into communities, are hard to find, especially for large scale networks. For this reason, we have performed our comparison over several synthetic datasets, estimating the resemblance of algorithms' communities with the provided ground truth partition of the network. As previously done in 7.1, to compare the ground truth with the structure delivered by the algorithm we adopt the Normalized Mutual Information (NMI) score, a measure of similarity borrowed from information theory [188] which in its general form is defined as:

$$NMI(X:Y) = \frac{H(X) + H(Y) - H(X,Y)}{(H(X) + H(Y))/2}$$

pdf where $H(X)$ is the entropy of the random variable X associated to the partition produced by the algorithm, $H(Y)$ is the entropy of the random variable Y associated to the ground truth



(a) CG (b) FB07 (c) WEIBO

(d) CG (e) FB07 (f) WEIBO

Figure 9.6: *(First row)* Community size distribution for CG (a), FB07 (b) and WEIBO (c). *(Second row)* Community per node distribution for CG (d), FB07 (e) and WEIBO (f).

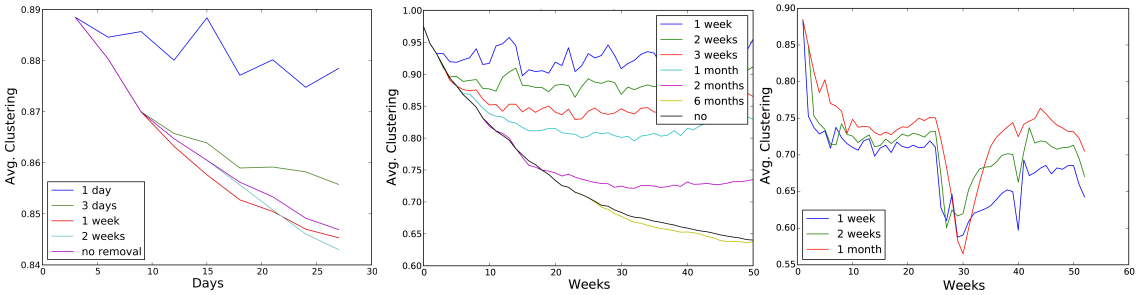
partition, whereas $H(X,Y)$ is the joint entropy. NMI ranges in [0,1] and is maximized when the confronted communities are identical. To cope with the overlapping nature of the extracted communities we adopt the NMI variant defined in [189].

To obtain a ground truth community partition of the networks we used the LFR benchmark [203], which generates synthetic networks along with ground truth communities, according to the following input parameters:

- N, the network size (from 1k to 500k nodes);
- C, the network density (from 0 to 0.9, steps of 0.1);
- μ, the average per-node ratio between the number of edges to its communities and the number of edges with the rest of the network (from 0 to 0.9, steps of 0.1).

We produced a total of 2500 different static synthetic networks varying the input parameters of the LFR model. On the generated networks we applied TILES and other overlapping community detection algorithms. One algorithm, iLCD [132] is a dynamic algorithm which re-evaluates communities at each new interaction produced by a streaming source according to the path lengths between each node and its surrounding communities. The other two algorithms are static ones: (i) DEMON [4], which exploits a label propagation procedure to build communities starting from ego networks; (ii) cFINDER [204], which computes communities based on the Clique Percolation Method (CPM). It is worth underlining that the LFR benchmark does not generate a timestamped stream of edges. For this reason, we imposed a random temporal order on the edges in order to simulate the streaming source S needed to apply TILES and iLCD. Conversely, such ordering is not needed for the other two algorithms since they are static.

As shown in Figure 9.5(a) we observe that varying the parameter μ, TILES produces communities whose NMI w.r.t. the ground truth is comparable to DEMON and cFINDER, but significantly



(a) CG (b) FB07 (c) WEIBO

Figure 9.7: Distribution of average clustering coefficient per community for CG (a), FB07 (b) and WEIBO (c). Each color identify a *ttl* value.

outperforms iLCD, its only direct competitor. In Figure 9.5(b) we can observe how the NMI of the compared methods are stable till the density parameter C reach 0.5 (half of all the possible edges are present in the network). Such a high density values, however, is not unusual for real interaction networks, where the density usually falls in the range $[0.05, 0.2]$. Figure 9.5(c) compares the execution time of Tiles, iLCD and Tiles$_p$, an instantiation of our algorithm which exploits the *compositionality* property in order to parallelize the computation. We can observe that the vanilla version of Tiles (implemented in Python) has an execution time comparable to iLCD (implemented in Java). Moreover, we are able to achieve the same results with Tiles$_p$ while significantly reducing the runtime. In the latter case we maintain the same interaction ordering used for the former approaches and impose limited parallelism, i.e., at most 2 synchronous updates at a time on consecutive interactions among nodes which satisfy the previously introduced constraint. Our algorithm produces communities whose NMI w.r.t. ground truth is significantly higher than the other dynamic community detection algorithm, with a similar execution time.

Evaluation on Real data

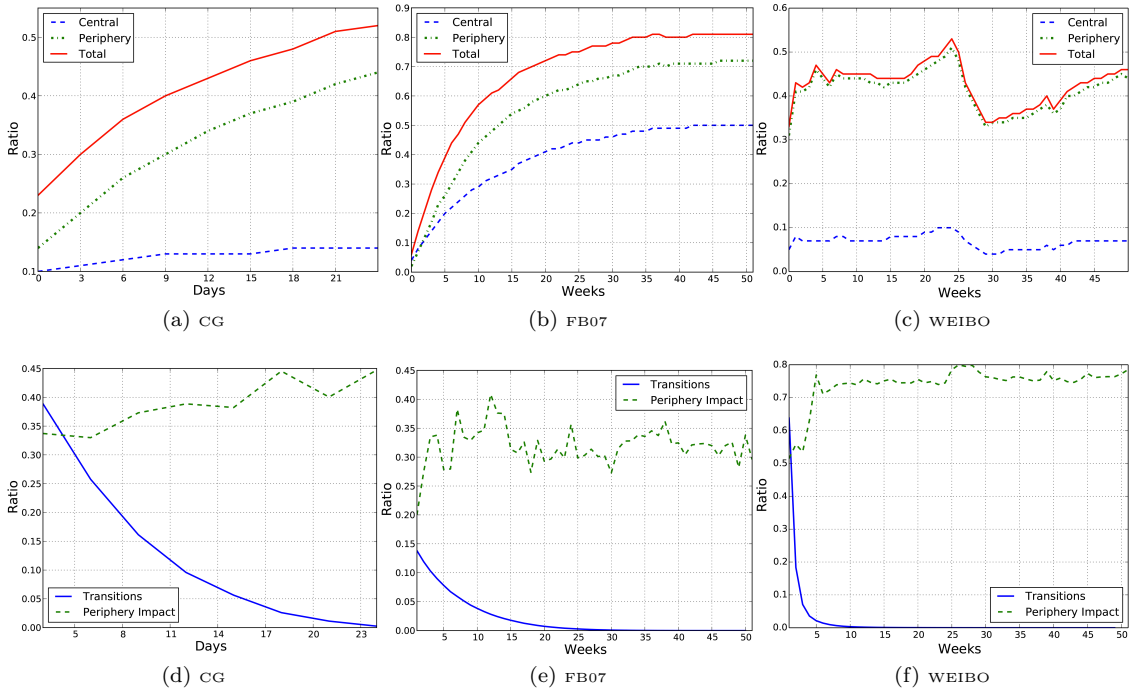
In order to characterize Tiles communities we analyzed three real world interaction networks: a wall post network extracted from Facebook, a Chinese micro-blogging mention network, and a nation-wide call graph extracted from mobile phone data. Those datasets allow us to test the algorithm on two different grounds: two "virtual" contexts where people share thoughts and opinions via social media platforms, and a "real" one where people directly keep in touch through a mobile phone. The general statistics of the datasets are shown in Table 9.1.

- **Facebook network**. The FB07 network used in our experiments is extracted from the WOSN2009 [205] dataset[3] and regards online interactions between users via the wall feature in the New Orleans regional network during 2007. We adopted an observation period τ of one week.

- **Call Graph**. The call graph is extracted from a big mobile phone dataset collected by a European carrier for billing and operational purposes. It contains date, time and coordinates of the phone tower routing the communication for each call and text message sent by $1,007,567$ costumers, in a period of one month. We discarded all the calls to external operators. As τ we adopted a window of 3 days.

- **WEIBO interactions**. This dataset is obtained from the 2012 WISE Challenge[4]: built upon the logs of a popular Chinese micro-blog service[5], its interactions represent mentions

[3] http://socialnetworks.mpi-sws.org/data-wosn2009.html
[4] http://www.wise2012.cs.ucy.ac.cy/challenge.html
[5] http://weibo.com



(a) CG (b) FB07 (c) WEIBO

(d) CG (e) FB07 (f) WEIBO

Figure 9.8: *(Top)* TILES communities nodes coverage; *(Bottom)* Distribution of transition time from periphery to core and ratio of nodes that move from the periphery to the core across consecutive observations.

of users in short messages. We selected a single year, 2012, and used an observation window one week.

It is worth noting that any arbitrary chosen value of τ does not affect the execution of TILES but only the moments of community status observation. The τ threshold is introduced with the mere purpose of simplifying the analysis of results and reduce the number of consecutive community observations. However, setting the τ parameter to the clock of the stream source will provide as output the full community updates history of communities.

We analyze four aspects of the communities produced by TILES: (i) the distribution of community size; (ii) the distribution of community overlap; (iii) the distribution of communities' average clustering coefficient; (iv) the transition time of nodes from the periphery to the core of communities. The size of TILES communities follows a heavy tail distribution for all datasets (Figure 9.6, top). This means that the vast majority of communities have few nodes while a small but significant portion of nodes have several thousands nodes. Such a great heterogeneity also characterizes the community overlap, i.e., how many different communities a node belongs to (Figure 9.6, bottom). The majority of nodes belong to just one or two communities, while some nodes belong to thousands different communities. Figure 9.7 shows the average clustering coefficient of communities computed over the core nodes. Communities maintain high average clustering coefficients as

Network	**Nodes**	**Edges**	**CC**	**#Observations (τ)**
FB07	19 561	304 392	0.104	52 (1 week)
CG	1 007 567	16 276 618	0.067	10 (3 days)
WEIBO	8,335,605	49,595,797	0.014	52 (1 week)

Table 9.1: Datasets statistics. *CC* identify the network clustering coefficient.

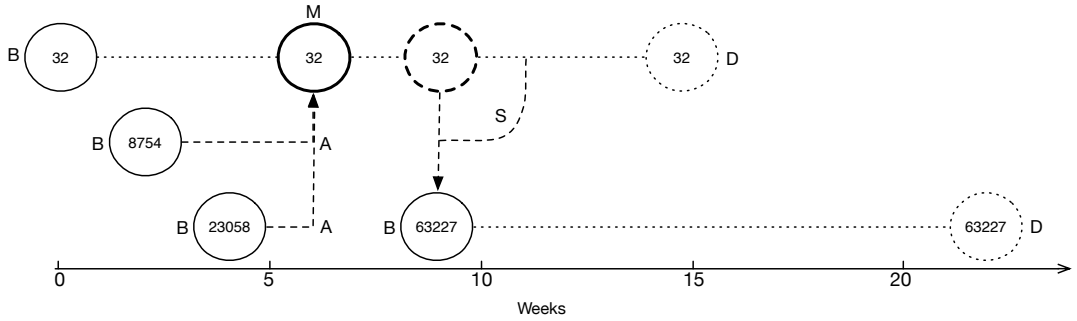


Figure 9.9: Example of Community Lifecycle extracted from WEIBO: each nodes represent a community with its id. Events are identified by the relative letter: (B) birth, (M) merge, (A) absorption, (S) split, (D) death. Merged communities and residual of splits are highlighted with thicker community lines.

time goes by, with minimum values of 0.6 for FB07 and WEIBO networks and 0.8 for CG. These values are significantly higher than the overall clustering coefficients of the networks (see Table 9.1) highlighting how Tiles is capable of identifies dense social structures. Due to its definition Tiles searches for precise patterns within a network structure: 3-clique based communities. The resulting nodes coverage, i.e., how many nodes are included into communities, is hence strictly related to the clustering coefficient of the analyzed network: the greater the clustering coefficient the higher is the nodes coverage. This tendency is depicted in Figure 9.8(top) where we report, for CG, FB07 and WEIBO, how the ratio of "core" nodes, "periphery" nodes and the sum of the two (total nodes coverage) change in time during the period of observation. FB07 has a clustering coefficient of 0.104 showing a high coverage: the 80% of nodes are included in some communities. In contrast, CG and WEIBO reach a coverage of $40 - 50\%$ due to their low overall clustering coefficient (0.067 and 0.014, respectively). A peculiarity of Tiles is the concept of community *periphery*. As already discussed, peripheral nodes are not involved in triangles with other nodes of the community. Every node first joins a community as peripheral node then it becomes core node once it is involved within a triangle with other core nodes. We find that the expected time of transition from the periphery to the core of a community is generally short (Figure 9.8 bottom, blue continuos line): in CG 40% of nodes become core nodes in just 3 days; in FB07 15% of nodes perform the transition during the first week; in WEIBO almost 60% of transitions occur within a single week. We also investigated how many nodes perform the transition from periphery to core across consecutive community observations: given two observation of a community C, what is the ratio of core nodes in C at $t + \Delta$ that where in the peripheral nodes at time t? We observe that this ratio has values between the 30% and 50% of the nodes for CG and FB07, and around 70% for WEIBO (Figure 9.8, green dashed line). This means that almost the half of the peripheral nodes become core nodes in the subsequent time window in all the networks.

Event-based Community Lifecycle

Several works on evolutionary community discovery focus on the analysis of the events which regulates community life-cycles (i.e., birth, merge, split and death of communities). Even if Tiles is not explicitly designed to output such events it allows us to identify them and capture the exact moments in which each event takes place. Observing step by step the network evolution we can track its perturbations. In order to perform an event-based analysis of community life-cycles we identify five main events:

- **Birth** (B): the community first appearance, this state coincides with the formation of the rising of the first set of core nodes;

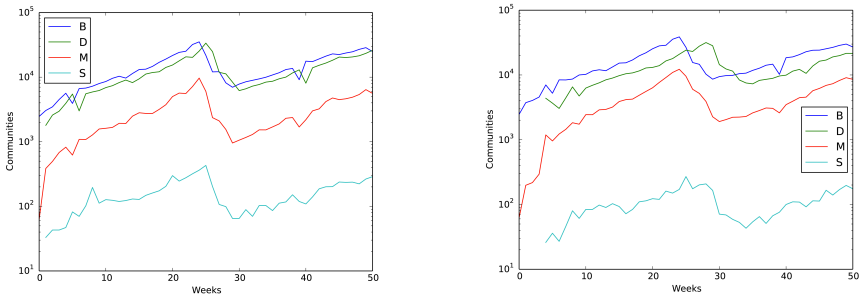
- **Merge/Absorption**: two (or more) communities merge when their core nodes completely overlap: we define Absorbed (A) the communities which expanding collide with an existing one and Merged (M) the already existing community;
- **Split** (S): a community splits in one or more subsets during the edge removal phase;
- **Death** (D): a community is considered dead when its core node set is empty.

In Figure 9.9 is shown an example of community life-cycle extracted from the WEIBO dataset. Analyzing the trends trough time of such events we can characterize the evolution of the whole network topology. As an example, Figure 9.10 shows such trends for WEIBO when the *ttl* is set to one week (left) and one month (right). We can notice that the observed trends follow more or less the same patterns regardless the chosen persistence threshold: the only major effect that we observe is the increase of merge and decrease of split events when a higher *ttl* is used.

Time To Leave analysis

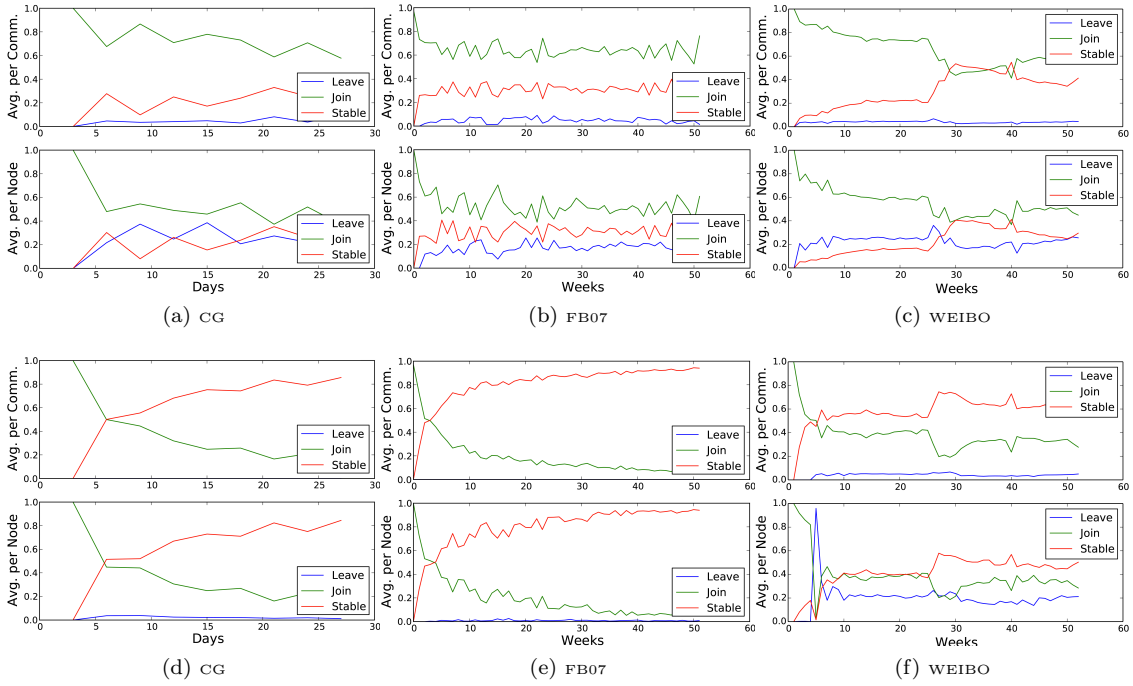
In order to analyze how different TTLs impact the communities characteristics we execute TILES varying the *ttl* values on FB07 (1 week, 2 weeks, 3weeks, 1 month, 2 months, 6 months $+\infty$), CG (1 day, 3 days, 1 week, 2 weeks, $+\infty$) and WEIBO (1 week, 2 weeks and 1 month). We have already shown how *ttl* affect the overall ratio of community life-cycle events, now we study the degree of stability induced on such structures at a micro level. We are interested in capturing the impact *ttl* has on the rates of node join/leave towards communities. Figure 9.11 shows two series of plots: (i) on the top, the trends for nodes' join/leave actions w.r.t. communities; (ii) on the bottom, how such trends relate on average to the stability of communities. As *ttl* increases, nodes and communities tend to stabilize quickly and *leave* actions emerge less frequently. On the other hand, Figure 9.12 shows the community average life (i.e., the number of weeks/days from its rising to its disappearance) w.r.t. its average size. A correlation between the two measures clearly emerges: in FB07 and CG bigger communities live longer regardless the value of *ttl* while in WEIBO we observe a negative correlation between size and average community life. Moreover, increasing *ttl* the expected life and size of communities tends to grow reaching their maximum when the removal phase is avoided ($ttl = +\infty$). This observation is reinforced by the average clustering coefficient trend shown in Figure 9.7: lower *ttl* values produce more compact community structures. Our experiments show that, as expected, interactions time to live deeply affects the outcome of the algorithm. In particular we observe that:

- higher *ttl* values produce bigger communities and foster the stabilization of the node memberships;
- lower *ttl* values produce smaller, denser, and often more unstable communities.



(a) One week removal (b) One month removal

Figure 9.10: Event trends in WEIBO. (a) One week vs. (b) one month edge removal.



(a) CG (b) FB07 (c) WEIBO

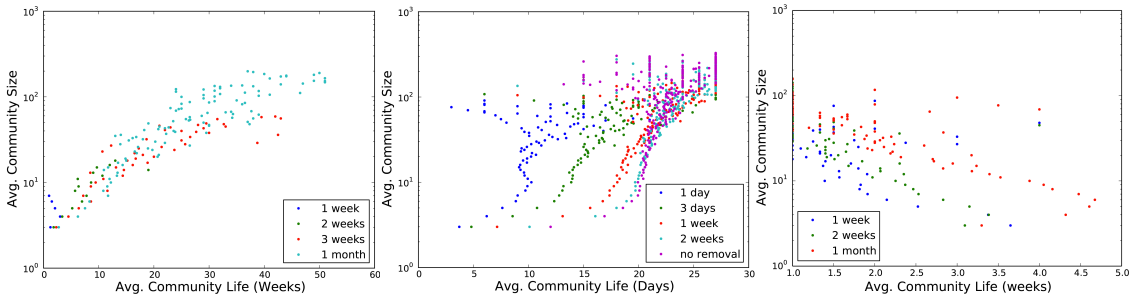
(d) CG (e) FB07 (f) WEIBO

Figure 9.11: Community membership evolution. *(First row)* Low TTL: CG one day, FB07 one weeks, WEIBO one week; *(Second row)* High TTL: CG and FB07 no removal, WEIBO one month.

We can argue that reasonable values for this threshold are the ones which lead to a stability rate trend (for both nodes and communities) that overcome the join/leave ones. When this condition is satisfied, TILES is able to extract communities with stable life-cycles (i.e., they do not appear and fall apart quickly). However, the choice of *ttl* is obviously context dependent (i.e., it is reasonable to assume that different phenomena can be characterized with different interactions persistence).

Discussion

In this work we proposed TILES, an algorithm which solves the problem of tracking the evolution of overlapping communities in interaction networks. TILES follows a "domino" approach: each new interaction determines the re-evaluation of community memberships for the endpoints and their neighborhoods. We define two types of community memberships: *weak membership*, describing nodes in the periphery of the community; and *strong membership*, for core nodes which are involved



(a) CG (b) FB07 (c) WEIBO

Figure 9.12: Community size vs. avg. life: each color identify a different *ttl* value.

in at least a triangle within the community. The *compositionality* is an interesting property of TILES, which allows for the parallelization of the algorithm and the speed up of communities computation. Other interesting characteristics, emerged by the application of the algorithm on real world scenarios (a call graph and two social interaction network), are the skewed distribution of community size, and the high average clustering coefficient within communities. Moreover, compared with another dynamic community detection algorithm on synthetic networks, TILES shows better execution times as well as a higher correspondence with the ground truth communities. Many lines of research remain open for future works: among them one of the most promising in our opinion involves the incorporation of dynamic aging procedures which automatically adjust the TTL of edges using information related to specific user trend activity. Moreover, the mechanisms which regulate the node transitions from the *periphery* to the *core* of a community are another interesting aspect to investigate: indeed, understanding them could lead to the extraction of very expressive features that can be used to approach the interaction prediction problem.

Chapter 10

Information Diffusion

> Diffusion is essentially a social process through which people talking to people spread an innovation.
>
> — *Everett Rogers*

In the previous chapters we have seen how network local structures as well as the complex topologies built upon them are subject to changes as time goes by. However, in social contexts, dynamism can be expressed not only by the edges and nodes which compose the network tissue but also by the information that flow through them. Several studies have addressed the problem of *Information Diffusion* from multiple perspectives with the aim of describe, model and forecast specific real world phenomena (i.e., virus spreading, information cascades...). In 10.1, starting from a dataset concerning the music listenings of almost 70 000 British users of Last.fm, a music-oriented OSN, we address a particular declination of the Information Diffusion problem: *Social Prominence*.

10.1 Social Prominence[1]

One of the most fascinating problems in SNA regards the understanding and modeling of diffusive phenomena. Modeling diffusion processes on complex networks enables us to tackle problems like: preventing epidemic outbreaks [152], favoring the adoption of new technologies and behaviors, design effective word-of-mouth communication strategies. In this work, we are focused on the social prominence aspect of the diffusion problem in networks.

In the setting of favoring social influence, most of the attention of researchers has been put on how to maximize the number of nodes subject to the spreading process. This is usually done by choosing appropriate seeds in critical parts of the network, such that their likelihood of being prominent users, i.e., nodes that are active on an innovation before all the other nodes, is maximum, to possibly achieve larger cascades. While larger cascades are obviously part of the overall aim, we argue that it is not the unique dimension of this problem. Three other dimensions are relevant: the *width*, the *depth* and the *strength* of the social prominence of any given node in a network. The width of a node is being prominent for its immediate neighbors; the depth capture its ability to be the root of long cascades; the strength is being the root of an intense activity.

Real-world scenarios focus on specific diffusion patterns requiring a multidimensional understanding of the prominence mechanics at play, along the three mentioned dimensions. Some examples are: (i) an analyst needs information from the personal acquaintances of a subject, the important aspect is that many subject's direct connections respond, ignoring people two steps away or more; (ii) a person wants to find another person with a given object, the important aspect is that some people are able to pass her message through a chain pointing to the target; (iii) an artist wants to influence people in a social network to her art, the important aspect is that some people are influenced above the threshold that will make them aware of the art. In (i) we want a broad diffusion in the first degree of separation. In (ii) we require a targeted diffusion similar to a Depth First Search. In (iii) there is the need of a high-intensity diffusion. Different scenarios may require any combination of the three.

In this work, we make use of three measures to capture the characteristics of these three scenarios: the Width, Depth and Strength of social prominence. The Width measures the ratio of the neighbors of a node that follows the node's actions. The Depth measures how many degrees of separation there are between a node and the other nodes that followed its actions. The Strength measures the intensity of the action performed by some nodes after the leader.

We study what the relations are between these three measures to understand if we are capturing three orthogonal dimensions of social prominence. We also study the relations between the Width, Depth and Strength measures and different node properties, with the aim of predicting the diffusion patterns of different events, given the characteristics of the nodes that lead their diffusion.

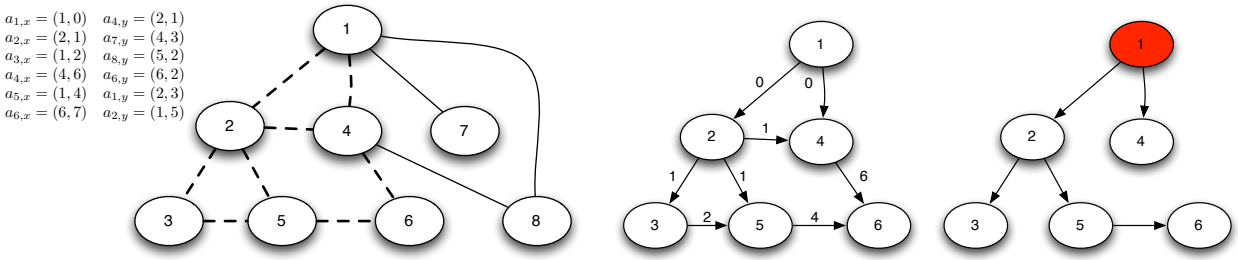
To validate our concepts, we constructed a social network from the music platform Last.Fm[2], along with the data about how many times and when each user listens to a song performed by a given artist. We detect who are the prominent users for each artist, i.e., the users who start listening to an artist before any of their neighbors. We calculate for each prominent user its Width, Depth and Strength, along with its network statistics such as the degree and the betweenness centrality, looking for associations between them. We then create a case study to understand what are the different dynamics in the spread of artists belonging to different music genres, by using the artists' tags.

10.1.1 Leader Characterization

Each diffusion process has its starting points. Any idea, disease or trend is firstly adopted by particular kinds of actors. Such actors are not like every other actor: they have an increased sensibility and they are the first to perform an action in a given social context. We call such actors prominent users, or *leaders*, because they are able to anticipate how other actors will behave. Given

[1] D. Pennacchioli, L. Pappalardo, G. Rossetti, D. Pedreschi, F. Giannotti, and M. Coscia, "The three dimensions of social prominence", in Social Informatics, 2013.

[2] `http://www.last.fm/`



(a) (b)

Figure 10.1: Toy Example. In (a) the social graph $\mathcal{G}$ and action set $\mathcal{A}$, where $x, y \in \Psi$ are the objects of the actions; in the *center* the induced subgraph for the action x; in (b) the diffusion tree for x. In red we highlighted the leader (root) for the given tree.

a graph, several interesting problems arise regarding how information spreads over its topology: can we identify the *leaders*? Can we characterize them? What kind of knowledge should we expect to extract from their analysis?

Our approach aims to detect *leaders* through the analysis of two correlated entities: the topology of the social graph and the set of actions performed by the actors (nodes). When discussing the roles of those entities, we refer respectively to the following definitions:

Definition 31 (Social Graph) *A social graph $\mathcal{G}$ is composed by a set of actors (nodes) V connected by their social relationships (edges) E. Each edge $e \in E$ is defined as a couple (u, v) with $u, v \in V$ and, where not otherwise specified, has to be considered undirected. With $\Gamma(u)$ we identify the neighbor set of a node u.*

Definition 32 (Action) *An action $a_{u,\psi} = (w, t)$ defines the adoption by an actor $u \in V$, at a certain time t, of a specific object ψ with a weight $w \in \mathcal{R}$. The set of all the actions of nodes belonging to a social graph $\mathcal{G}$ will be identified by $\mathcal{A}$, while the object set will be called Ψ.*

We identify with $\mathcal{G}_\psi = (V_\psi, E_\psi)$, where $V_\psi \subset V$ and $E_\psi \subset E$, the induced subgraph on $\mathcal{G}$ representing respectively the set of all the actors that have performed an action on ψ, and the edges connecting them. We depict an example of the social graph and the set of actions in Figure 10.1 *(a)*, where the induced subgraph for the object x is highlighted with a dashed line. In the Figure, $a_{1,x}$ refers to the user 1 performing the action x; and $a_{1,x} = (1, 0)$ means that user 1 performed x one time, starting at the tilmestep 0.

Given the nature of a diffusion process, we would expect that each *leader* will be prominent among its neighbors, being the root of a cascade event that follows some rigid temporal constraints. Our constraint is that a node u precedes a neighbor v iff given $t_{u,\psi} \in a_{u,\psi}$ and $t_{v,\psi} \in a_{v,\psi}$ is verified that $t_{v,\psi} > t_{u,\psi}$ and $t_{v,\psi} - t_{u,\psi} \leq \delta$. Here, δ is a temporal resolution parameter that limits the cascade effect: if $t_{v,\psi} - t_{u,\psi} > \delta$, we say that v executed action $a_{v,\psi}$ independently from u, as u's prominence interval is over.

We transform each undirected subgraph $\mathcal{G}_\psi$ in a directed one imposing that the source node of an edge must have performed its action before the target node. After that, each edge (u, v) will be labeled with $min(t_{u,\psi}, t_{v,\psi})$ to identify when the diffusion started going from one node to the other. The directed version of $\mathcal{G}_\psi$ represent all the possible diffusion paths that connect leaders with their "tribes" (Figure 10.1 *(b)* an example for the object $x \in \Psi$).

From now on, for a given object ψ, we will refer to the corresponding leader set as $\mathcal{L}_\psi$: when no action is specified the set $\mathcal{L}$ will be used to describe the union of all the $\mathcal{L}_\psi$ for the graph $\mathcal{G}$. To be defined a *leader* an actor should not have any incoming edges in $\mathcal{G}_\psi$. This is because a prominent user cannot act after another user (they are, in their surroundings, innovators), and is a direct consequence to the adoption of a directed graph to express diffusion patterns. Given this definition, for each directed connected component $\mathcal{C}_\psi \subset \mathcal{G}_\psi$ multiple nodes can belong to $\mathcal{L}_\psi$.

Algorithm 10 The pseudo-code of ExtractLeaders.

Require: $\mathcal{G} = (V, E); \mathcal{A}; \Psi, \delta;$
Ensure: $\mathcal{L}, \mathcal{T}$

$\mathcal{T} \leftarrow \{\}$, $\mathcal{L} \leftarrow \{\}$
for $\psi \in \Psi$ **do**
  $\mathcal{G}_\psi \leftarrow$ InducedSubgraph$(\mathcal{G}, \psi, \delta)$
  $\mathcal{T}_\psi \leftarrow \{\}$, $\mathcal{L}_\psi \leftarrow \{\}$
  for $\mathcal{C}_\psi \in \mathcal{G}_\psi$ **do**
    for $l \in \mathcal{C}_\psi$ **do**
      if InDegree$(\mathcal{C}_\psi, l)$==0 **then**
        $\mathcal{L}_\psi \leftarrow \mathcal{L}_\psi \cup l$
        $\mathcal{T}_{l,\psi} \leftarrow MST(C_\psi, l)$
        $\mathcal{T}_\psi \leftarrow \mathcal{T}_\psi \cup \mathcal{T}_{l,\psi}$
      end if
    end for
  end for
  $\mathcal{L} \leftarrow \mathcal{L} \cup \mathcal{L}_\psi$
  $\mathcal{T} \leftarrow \mathcal{T} \cup \mathcal{T}_\psi$
end for
return $\mathcal{L}, \mathcal{T}$

Realistically, a leader may be influenced by exogenous events. This is not a problem as we are not measuring a node's influence, but a node's prominence, i.e., its propensity to act faster than others to any kind of exogenous and/or endogenous influence. To study the path of diffusion given an action a and a leader l we use a minimum diffusion tree:

Definition 33 (Leader's Minimum Diffusion Tree) *Given an action a_ψ, a directed connected component $\mathcal{C}_\psi$ and a leader $l \in \mathcal{L}_\psi$, the minimum diffusion tree $T_{l,\psi} \subset \mathcal{C}_\psi$ is the* Minimum spanning tree *(MST) having its root in l and built minimizing the temporal label assigned at the edges.*

An example of minimum diffusion tree for the node 1 and object x is shown in Figure 10.1 *(right)*. For each object, the diffusion process on a given network is independent. Moreover, given temporal dependencies on its adoption (expressed through actions $a_{*,\psi} \in \mathcal{A}$), it is possible to identify the origin points of the diffusion. The identified *leaders* will show different topological characteristic and will be prominent in their surroundings in different ways: our aim is to classify diffusion *leaders* characterizing some of their common traits.

To sum up, we use the leader extraction procedure ExtractLeaders as defined in Algorithm 10. For all objects $\psi \in \Psi$, we extract the directed induced subgraph $\mathcal{G}_\psi$ by filtering all nodes that performed an action $a_{*,\psi} \in \mathcal{A}$ and all the edges between them (performed by InducedSubgraph where δ is the temporal constraint discussed before). Then, for each connected component $\mathcal{C}_\psi \in \mathcal{G}_\psi$ we choose as our leaders all the nodes without incoming edges. We add them to the leader set $\mathcal{L}_\psi$ and we store in $\mathcal{T}_\psi$ their Minimum Diffusion Trees (calculating the minimum spanning tree MST with root in l using only nodes in C_ψ). At the end, we return the union of $\mathcal{L}_\psi$ and $\mathcal{T}_\psi$.

The proposed algorithm has low complexity. First, we cycle over all the actions ($\mathcal{O}(|\Psi|)$). Then, we cycle over all the connected components of $\mathcal{G}_\psi$ and, for each one, we cycle over the nodes belonging to them: together the two cycles reach the complexity of $\mathcal{O}(|V_\psi|)$. Within the inner loop a minimum spanning tree ($\mathcal{O}(\log |V|)$, with Kruskal's algorithm) is computed for every leader. As a consequence, the final complexity of ExtractLeaders is $\mathcal{O}(|\Psi| \times |V| \log |V|)$. For large networks, it is fair to assume that $|\Psi| << |V|$, so the complexity would be $\Theta(|V| \log |V|)$. Moreover, since each action is independent from the others, with $|\Psi|$ processors the exact complexity would be $\mathcal{O}(|V| \log |V|)$.

10.1.2 Local Diffusion Measures

To capture the three dimensions of social prominence we need three network measures. We call these measures *Width*, the ratio of neighbors mirroring an action after a node; *Depth*, how many degrees of separation are in between a node and the most distant of the nodes mirroring its actions; and *Strength*, how strongly nodes are mirroring a node's action.

Given a leader, the Width aims to capture the direct impact of her actions on her neighbors, i.e., the degree of importance that a leader has over her friends.

Definition 34 (Width) *Let G be a social graph, $\psi \in \Psi$ an object and $l \in \mathcal{L}_\psi \subset V$ a leader: the function $width : \mathcal{L}_\psi \to [0,1]$ is defined as:*

$$width(l,\psi) = \frac{|\{u | u \in \Gamma(l) \wedge \exists a_{u,\psi} \in \mathcal{A}\}|}{|\Gamma(l)|} \tag{10.1}$$

The value returned is the ratio of all the neighbors that, after the action of the leader, have adopted the same object.

The Depth measure evaluates how much a leader can be prominent among other prominent leaders, which can be prominent on other leaders and so on.

Definition 35 (Depth) *Let $T_{l,\psi}$ be a minimum diffusion tree for a leader $l \in \mathcal{L}_\psi$ and a given object $\psi \in \Psi$: the function $depth : T_{l,\psi} \to \mathbb{N}$ computes the length of the maximal path from l to a node $u \in T_{l,\psi}$. The function $depth_{avg} : T_{l,\psi} \to \mathbb{R}$ computes the average length of paths from l to any leaf of the tree.*

The last proposed measure, the Strength, tries to capture quantitatively the total weight of the adoption of an object after the leader's action. A leader is strongly prominent if the nodes among which she is prominent are very engaged in adopting what she adopted. Direct prominence diminishes as new adopters become more distant, in the network sense, from the original innovator. Therefore, we decided to introduce a distance damping factor.

Definition 36 (Strength) *Let $T_{l,\psi}$ be a minimum diffusion tree for a leader $l \in \mathcal{L}_\psi$ and an object $\psi \in \Psi$; $0 < \beta < 1$ a damping factor: the function $strength : T_{l,\psi} \times (0,1) \to \mathbb{R}$ is defined as:*

$$strength(T_{l,\psi}, \beta) = \sum_{i \in [0, depth(l)]} \beta^i L(T_{l,\psi}, i) \tag{10.2}$$

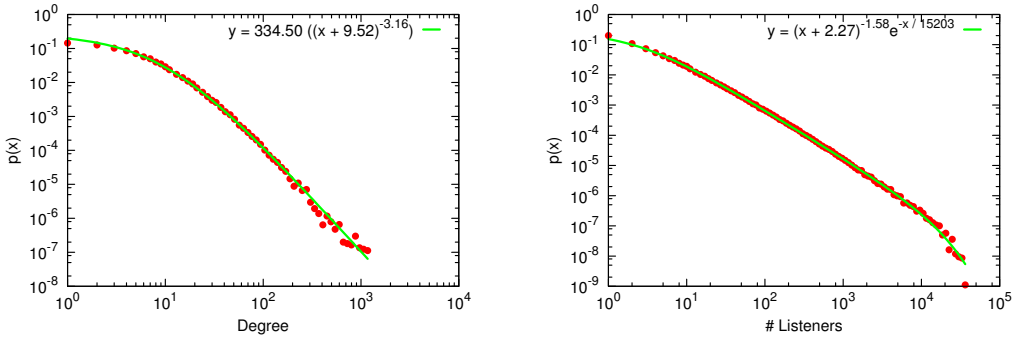
where $L : T_{l,\psi} \times \mathbb{N} \to \mathbb{R}$ is defined as:

$$L(T_{l,\psi}, i) = \sum_{\{u | u \in T_{l,\psi} \wedge distance(l,u) = i\}} \frac{w_{u,\psi}}{w_u} \tag{10.3}$$

and represents the sum, over all the nodes u at distance i from l, of the ratio between the weight of action ψ and the total weight of all the actions taken.

Example 4 *Given the graph in Figure 10.1, what are the Width, Depth and Strength values for the red node leader and the action x?*

- ***Width**: from Figure 10.1*(left) *we see that $\Gamma(1) = \{2,4,7,8\}$, i.e., 4 nodes. Given that $\Gamma_x(1) = \{u | u \in \Gamma(1) \wedge \exists a_{u,x}\} = \{2,4\}$, we have $width(1,x) = \frac{|\Gamma_x(1)|}{|\Gamma(1)|} = 0.5$.*
- ***Depth**: the leaves in Figure 10.1*(rigth) *are nodes 3, 4 and 6. Node 4 is a direct neighbor of 1, while node 3 is two edges away. The longest chain is $1 \to 2 \to 5 \to 6$, therefore $depth(T_{1,x}) = 3$. We can also calculate $depth_{avg}(T_{1,x})$, that is the average path length in the tree from node 1 to all the leaves: $\frac{1+2+3}{3} = 2$.*



(a) (b)

Figure 10.2: *(a)* Log-binned distribution of the nodes' degree. *(b)* Log-binned distribution of number of listeners per artist.

- ***Strength**: we need to use the number of times each node performed action x. We also set our damping faction $\beta = 0.5$. At the first degree we have nodes 2 and 4, that performed action x 2 and 4 times respectively; they also performed action y 1 and 2 times respectively: their contribution is then $\beta^0 \times (\frac{2}{2+1} + \frac{4}{4+2})$. Nodes 2 and 5 are at the second degree of separation as they never performed action y, therefore they add: $\beta^1 \times (1+1)$. Finally, at the third degree of separation, node 6 adds $\beta^2 \times \frac{6}{6+6}$. Wrapping up, strength$(T_{1,x}, 0.5) = 2.458\bar{3}$.*

Experiments

Here we present the analyzed data which were extracted from the music OSN Last.fm. We use the data to characterize the Width, Depth and Strength measures, by searching for associations with network topology measures. Finally, we analyze the prominence of different users for different musical genres.

Last.fm Data

Last.fm is an online social network platform, where people can share their own music tastes and discover new artists and genres basing on what they, or their friends, like. Users send data about their own listenings. For each song, a user can express her preferences and add tags (e.g., genre of the song). Lastly, a user can add friends (undirected connections, the friendship request must be confirmed) and search her neighbors w.r.t. musical tastes. A user can see, in her homepage, her friends' activities. The co-presence of these characteristics makes Last.fm the ideal platform on which test our method, as it contains everything we need: social connections that can convey social prominence, a measure of intensity proportional to the number of listening of an artist, rich metadata attached to each song/artist and an intrinsic temporal dimension of users' actions.

Using Last.fm APIs[3], we obtained a sample of the UK user graph, exploring the network with a breadth-first approach, up until the fifth degree of separation from our seeds. For each user, we retrieved: (a) her connections, and (b) for each week in the time window from Jan-10 to Dec-11, the number of single listenings of a given artist (e.g., in the week between April 11,2010 and April 18,2010 the user 1234 has listened 66 songs from the artist Metallica).

For each artist we have a list of tags, weighted with the number of users that assigned the tag to the artist (e.g., Metallica has 4 tags: "metal" with counter 50 670, "hard rock" with 23 405, "punk" with 10 500 and "adrenaline" with 670). We split tags, associating the counter to each single word (in the last example: (metal, 50 670), (punk, 10 500), (hard, 23 405), (rock, 23 405), (adrenaline, 670)), then we filtered the words referring to a musical genre ((metal, 50 670), (punk, 10 500), (rock,

[3] `http://www.last.fm/api/`

	Width	Strength	Degree	Clustering	Neigh Deg	Bet Centr	Clo Centr
AVG Depth	-0.03	**-0.23**	-0.08	0.05	-0.08	-0.02	**-0.13**
Width	-	0.01	**-0.31**	**0.13**	0.05	-0.07	**-0.59**
Strength	-	-	0.02	-0.02	0.03	0.00	0.04
Degree	-	-	-	**-0.16**	-0.02	**0.77**	**0.56**
Clustering	-	-	-	-	-0.05	-0.06	**-0.32**
Neigh Deg	-	-	-	-	-	-0.00	**0.39**
Bet Centr	-	-	-	-	-	-	**0.22**

Table 10.1: Pearson correlation coefficient ρ between Width, Depth, Strength and other network statistics for our leaders.

23 405)). Finally, we assigned a musical genre to an artist iff the survived tag with the greater counter had the relative rate ≥ 0.5 (in the example: $r_{metal}(Metallica) = \frac{50670}{50670+10500+23405} \simeq 0.6$, so Metallica are definitely metal).

After the crawl and cleaning stages, we built our social graph $\mathcal{G}$. In $\mathcal{G}$ each node is a user and each edge is generated using the user's friends in the social media platform. The total amount of nodes is 75969, with 389639 edges connecting them. In Figure 10.2(a) we depicted the log-binned degree distribution of $\mathcal{G}$, along with the best fit. Each action in the data is one user listening to an artist w times in week t. In Figure 10.2(b) we depicted the log-binned distribution of the number of listeners per artist, along with the best fit.

Since we are interested in leaders, we need to focus only on new artists that were previously not existent. If an artist was in activity before our observation time window, there is no way to know if a user has listened to it before, therefore nullifying our leader detection strategy. For this reason, we focus only on artists whose first listening is recorded six months after the beginning of our observation period. Each artist belongs to a music genre (coded in its tag). We decided to focus on music genres with sufficient popularity, namely: dance, electronic, folk, jazz, metal, pop, punk, rap and rock. A genre's popularity is determined by having at least 10 artists with at least 100 listeners. To sum up, we focus on the artists who appear for the first time after six months in our observation period, with at least 100 listeners and belonging to one of the mentioned nine tags. The cardinality of our action set $\mathcal{A}$ is 168216 actions, while the object set Ψ contains a total of 402 artists.

In our experimental settings, we set our damping factor $\beta = 0.5$ for the calculation of the Strength measure. We also set $\delta = 3$, meaning that if a user listened to a particular artist three weeks or more after its neighbor then we do not consider her neighbor to be prominent for her for that action.[4]

Characterization of the Measures

For each leader, besides Width, Depth and Strength, we calculated also the degree (number of edges connected to the node), the clustering coefficient (ratio of triangles over the possible triads centered on the node), the neighbor degree (average degree of the neighbors of the node), the betweenness (share of the shortest paths that pass through the node) and closeness centrality (inverse average distance between the node and all the other nodes of the network).

In Table 10.1 we report the Pearson correlation coefficient ρ between the network measures. We highlighted the correlations whose p-value was significant or whose absolute value was strong enough to draw some conclusions. For the significance of p-values, the traditional choice is to set the threshold at $p < 0.01$. However, given the high number of observations available, we decided to be more restrictive, setting our threshold at $p < 0.0005$. We also consider a ρ value significant if $|\rho| > 0.1$.

The Depth measure is associated with low closeness centrality. This means that a deep prominence is associated to nodes at the margin of the network. It is expected that nodes with high

[4]To assure experiment repeatability, we made our cleaned dataset and our code available at the page `http://goo.gl/h53hS`

	Clustering	Clo Centr
Partial ρ	0.087216	-0.536861
p-value	1.57×10^{-14}	0

Table 10.2: Partial correlation and p-value of Clustering and Closeness Centrality with Width, controlling for Degree values.

closeness centrality have also low Depth: being central, they cannot generate long chains of diffusion. The eccentricity of all the nodes of the network ranges from 6 to 10, meaning that some leaders cannot have a Depth larger than 5. To make a fair comparison, we recalculate the Depth value capping it at 5, meaning that any Depth value larger than 5 is manually reduced to 5. Then, we recalculate the correlation ρ between the Depth capped to 5 and the closeness centrality obtaining as result $\rho = -0.1366$, with $p < 0.0005$. We can conclude that central nodes are not associated with deep spread of their prominence in a social network.

For the Width measure, the anti-correlation with the degree is not meaningful, as the degree is in the denominator of Definition 34. However, we observe a positive association with clustering, i.e., nodes could be prominent in a tightly connected community; and a negative association with closeness centrality, i.e., central nodes could not spread a wide influence. Both associations could be explained with the negative correlation with degree. Therefore, for both measures we run a partial correlation, controlling for the degree. In practice, we calculate the correlation between Width and clustering (or closeness centrality) by keeping the degree constant. Results are in Table 10.2: even if significant according to the p-value, the relationship between Width and clustering is very weak and deserves further investigation. On the other hand, it is confirmed that central nodes are also associated with low Width, regardless their degree.

From Table 10.1, we see that the Strength measure is not correlated with traditional network statistics. As a consequence, hubs associated with low Depth and low Width, do not have necessarily high Strength, making their prominence in a network questionable. Moreover, Strength appears to be negatively associated with Depth, suggesting a trade-off between how deeply a node can be prominent in a network and how strong this prominence is on the involved nodes.

The anti-correlation between the Strength and the Depth may be due to β: from Definition 36 β decreases nodes' contributions at each degree of separation (i.e., at increasing Depths). As a consequence, nodes farther from the leader contribute less to its Strength, i.e., the highest the Depth the smallest are the contributions to the Strength. We recalculated the Strength values by setting $\beta = 1$, therefore ignoring any damping factor and nullifying this effect. We obtained as result $\rho = -0.4168$ and a significant p-value, therefore concluding that β is not causing the anti-correlation between Depth and Strength.

To sum up, we summarize the associations as follows: (i) central nodes are not necessarily prominent in a social network (low Width and Depth), a result that confirms [206] and [207]; (ii) longer cascades (higher Depths) are associated with a lower degree of engagement (lower Strengths), a phenomenon possibly related to the role played by "weak ties"; (iii) be prominent among neighbors is probably easier if the node is in a tightly connected community, but more evidences have to be brought to reject the role played by the node's degree.

Case Study: Music Genres

Here, we present a case study based on Last.fm data. Our aim is to use our Leader extraction technique and the proposed Width, Depth and Strength measures to characterize the spread of musical genres among the users of the service. We recall that the object set Ψ is composed by 402 artists, each one having a tag corresponding to her main music genre.

For each couple leader l and object ψ, we calculate Depth, Width and Strength values; we compute the size of the Leader's Minimum Diffusion Tree ($|T_{l,\psi}|$); and we group together the objects with the same tag. To characterize the typical values of Width, Depth and Strength for each tag we cannot use the average or the median. This is because Strength and Width values are skewed, and it is the combination of the three measures that really characterizes the leaders.

Cluster	size	dance	ele	folk	jazz	met	pop	punk	rap	rock
0	1822	1.25	1.13	**1.54**	1.37	1.50	0.76	1.31	1.13	1.10
1	136	1.28	1.55	1.28	**2.35**	0.78	0.73	0.64	1.35	0.70
2	664	0.59	0.87	0.98	0.48	0.95	0.97	**1.50**	1.20	1.19
3	482	1.26	1.16	1.09	1.12	0.91	0.80	**2.48**	1.24	0.89
4	973	1.14	1.20	1.15	**1.41**	0.80	0.91	0.66	0.97	0.97
5	512	**1.29**	0.96	0.95	1.09	1.10	0.97	0.33	1.06	1.01
6	682	0.89	0.79	0.61	0.64	**1.13**	1.08	1.07	1.08	1.01
7	124	0.75	**1.45**	0.35	0.64	0	1.09	0	1.02	0.62
8	524	0.93	1.01	1.12	0.91	**1.15**	1.07	0.43	0.95	0.87
9	937	0.40	0.46	0.19	0.23	0.45	**1.56**	0.13	0.37	1.06
10	232	0.72	0.57	0.27	0.99	0.38	**1.44**	0.38	0.46	1.00
11	612	0.74	0.94	0.71	0.40	0.70	**1.27**	0.07	0.68	0.83

Table 10.3: (a) The RCA scores of the presence of each tag in each cluster.

We cluster leaders using as features their Width, Depth and Strength values. We used the Self-Organizing Map (SOM) method [208] because:

- SOM does not require to set the number of clusters k;
- k-means outperforms SOM only if the number of resulting clusters is very small (less than 7) [209], but our study of the best k to be used in k-means with the Sum of Squared Errors (SSE) methodology resulted in an optimal number of clusters falling in a range between 9 and 13 (in fact, SOM returned 12 clusters); and
- SOM performs better if the data points are contained in a warped space [210], which is our case.

In Table 10.3, we report a presence score for each tag in each cluster. There are larger and smaller clusters and some tags attract more listeners than others. To report just the share of leaders with a given tag in a given cluster is not meaningful. We correct the ratio with the expected number of leaders with the given tag in the cluster, a measure known as Revealed Comparative Advantage: $RCA(i,j) = \frac{freq_{i,j}}{freq_{i,*}} \Big/ \frac{freq_{*,j}}{freq_{*,*}}$, where i is a tag, j is a cluster, $freq_{i,j}$ is the number of leaders who spread an artist tagged with tag i that is present in cluster j. For each cluster we highlighted the tag with the highest unexpected presence.

The centroids of the SOM are depicted in Figure 10.3: Depth on the x-axis, Strength on the y-axis and the Width as the color (Strength and Width are in log scale). We can identify the clusters characterized by the highest and lowest Strength (9 and 4 respectively); by the highest and lowest

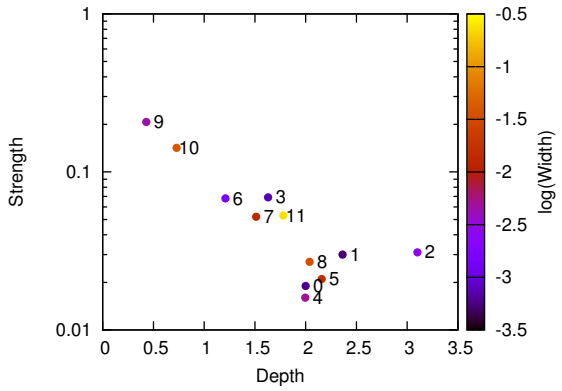


Figure 10.3: The centroids of our clusters.

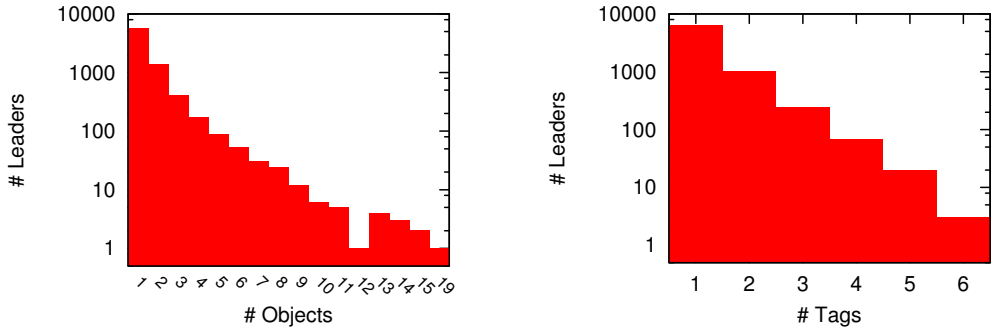


Figure 10.4: Distribution of number of objects *(a)* and of tags *(b)* per leader.

Depth (2 and 9 respectively); and by the highest and lowest Width (11 and 1 respectively). There are also clusters with relatively high combinations of two measures: cluster 10 with high Strength and Width or cluster 5 with high Depth and Width.

From Table 10.3 we obtain a description of what values of Width, Depth and Strength are generally associated with each tag. For space constraints, we report only a handful of them for the clusters with extreme values. Jazz dominates clusters 1 (with the lowest Width) and 4 (with the lowest Strength): this fact suggests that jazz is a genre for which it is not easy to be prominent.

Cluster 9, with the lowest Depth but the highest Strength, is dominated by pop (that dominates also clusters 10 and 11, both with high Strength but low Depth). As a result, we can conclude that prominent leaders for pop artists are embedded in groups of users very engaged with the new artist. On the other hand, it is unlikely that these users will be prominent among their friends too.

Finally, cluster 2 with the highest density has a large majority of punk leaders. From this evidence, we can conclude that punk is a genre that can achieve long cascades, exactly the opposite of the pop genre.

We move on to the topological characteristics of the leaders per tag. A caveat: a leader is not bounded to be leader just for one object ψ, but she is free to be prominent in many ψ. Thus, one leader can be counted in more than one tag. To help understand the magnitude of the issue, we depicted in Figure 10.4 the number of leaders influencing their neighbors for a given amount of actions (left) and for a given amount of tags (right). The y axis is logarithmic. The typical leader influences one neighbor for one artist. However, some leaders express their leadership for 8 objects and 4 tags.

In Figure 10.5 we depict the log-binned distributions, for the leaders of each tag, of four topological measures: degree, closeness centrality, clustering and neighbor degree. We omit betweenness centrality for its very high correlation with degree. Overall, there is no significant distinction between the tags in the distributions of the topological features.

The most noticeable information is carried by the degree distributions (Figure 10.5(a)). Each distribution appears very different from the overall degree distribution (Figure 10.2(a)). There are fewer leaders with low degree than expected, therefore it appears that a high degree increases the probability of being a leader. On the other hand, we know that central hubs have on average lower Depth and Width. As a consequence, it appears that the best leader candidates are the nodes with an average degree, and from Figure 10.5(a) we see that each tag has many leaders with a Degree between 10 and 100.

Using our leaders' Minimum Diffusion Trees, we extract some patterns that help us obtaining a complementary point of view over the leader prominence for different music genres. We mine a graph dataset composed by all diffusion trees $T_{l,\psi}$ with the VF2 algorithm [211]. Suppose we are interested in counting how frequent is the following star pattern: a leader influences three of its neighbors in the diffusion trees of pop artists. In our data, we have 5043 diffusion trees for pop

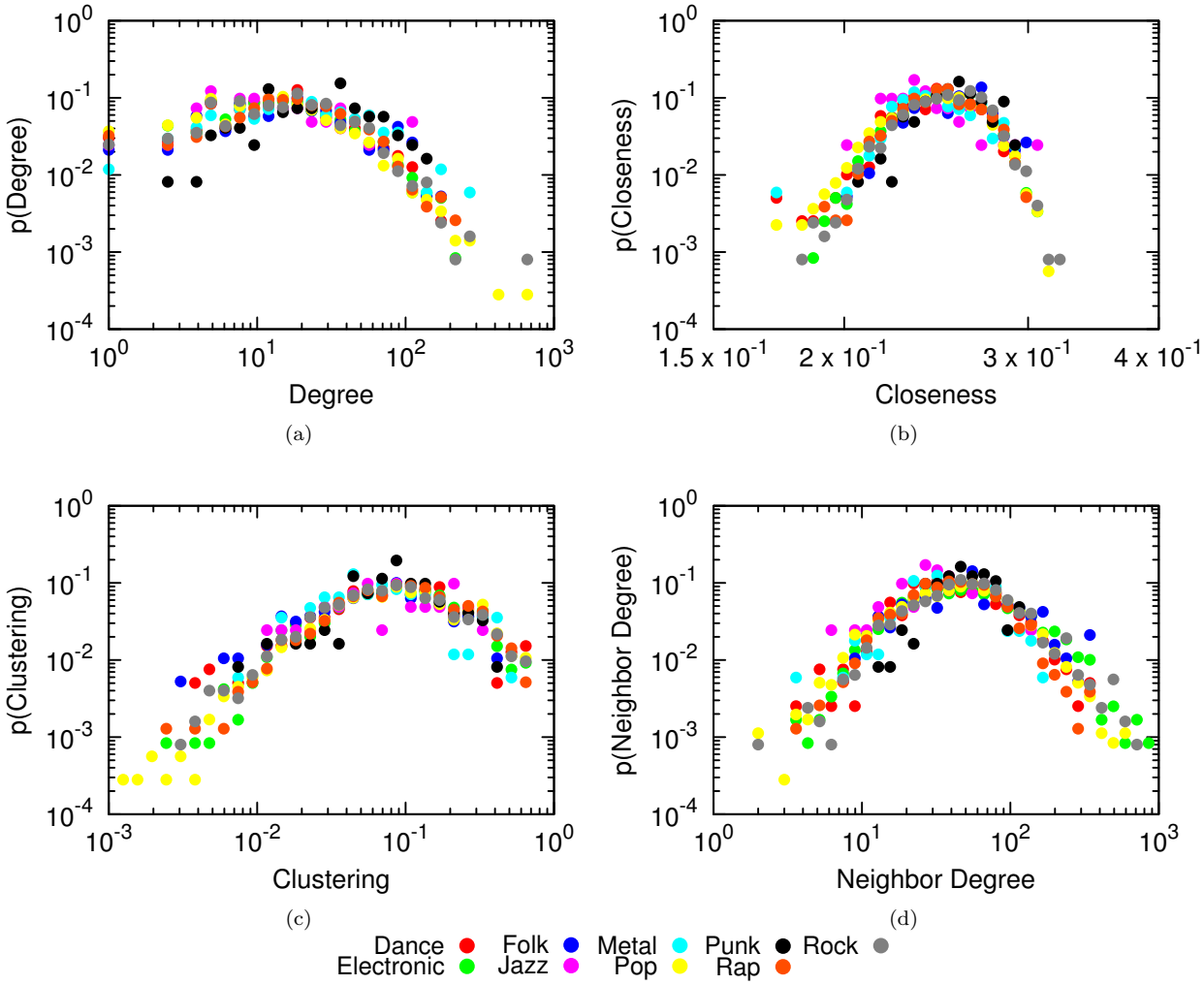


Figure 10.5: Distribution of leaders' Degree *(a)*, Closeness Centrality *(b)*, Clustering *(c)* and Neighbor Degree *(d)* per tag.

Pattern	dance	electronic	folk	jazz	metal	pop	punk	rap	rock
	3.62% (35.42%)	3.04% (22.50%)	3.94% (30.30%)	7.25% (62.50%)	4.14% (23.08%)	3.69% (32.01%)	6.56% (27.59%)	4.01% (27.97%)	4.22% (30.43%)
	2.55% (25.00%)	3.92% (29.00%)	3.15% (24.24%)	4.35% (37.50%)	4.83% (26.92%)	3.61% (31.29%)	10.66% (44.83%)	5.60% (38.98%)	4.12% (29.71%)
	3.40% (33.33%)	3.79% (28.00%)	3.94% (30.30%)	4.35% (37.50%)	6.90% (38.46%)	4.73% (41.01%)	12.30% (51.72%)	4.99% (34.75%)	4.52% (32.61%)

Table 10.4: Presence of different diffusion patterns per tag.

artists, and 581 have at least four nodes. Since the VF2 algorithm found the star pattern in 186 of these graphs, we say that it appears in 3.69% of the trees, or in 32.01% of the trees that have enough nodes to contain it.

In Table 10.4 we report the results of mining three patterns of four nodes: i) the star-like pattern described above; ii) a chain where each node is prominent for (at least) one neighbor; iii) a split where the leader is prominent for a node, which itself is prominent for two other neighbors. Two values are associated to each pattern and tag pair: the relative overall frequency, and the relative frequency considering only the trees with at least four nodes (in parentheses).

There is no necessary relation between the patterns and Width, Depth and Strength measures: a low Depth does not imply the absence of the chain pattern, nor does a high Width imply a high presence of the star pattern. However, the combination of the two measures may provide some insights. For instance, we saw in Fig 10.3 that jazz leaders are concentrated in the lowest Width cluster. However, many jazz leaders who affect at least three nodes tend to be prominent in their neighbors, much more than in any other genre (7.25% of all leaders, 62.5% of leaders who are prominent for at least three other nodes). Therefore, jazz leaders have low prominence among their friends, however they are likely to have at least three neighbors for which they are prominent.

The chain pattern is more commonly found in pop leaders than in folk ones, even though the clusters of their leaders described in Table 10.3 would suggest the opposite. It seems that pop leaders are not likely to be prominent for nodes any further than the third degree of separation, while folk leaders tend to generate longer cascade chains. Also in this case, punk leaders are commonly found in correspondence with chain patterns, just as Table 10.3 suggested.

Although pop leaders show a much greater Strength value than metal ones (by confronting in Figure 10.3 their presence in high Strength clusters like 9 or 10 and low Strength clusters like 8 and 0), the split pattern tends to be more frequent in the metal genre (6.90% against 4.73% of the trees). This phenomenon suggests us that metal leaders tend to be prominent for nodes strongly devoted to metal, inducing them to spread the music to their neighbors. Pop leaders, on the other hand, affect more neighbors with higher Width and Strength, presumably flooding their ego networks with the songs they like.

Discussion

We presented a study of the propagation of behaviors in a social network. Instead of just studying cascade effects and the maximization of influence by a given starting seed, we decided to analyze three different dimensions: the prominence of a leader w.r.t. its direct neighbors, w.r.t. the distance it spread its influence and w.r.t. the level of engagement shown by influenced nodes. We characterized each of these concepts with a different measure: Width, Depth and Strength. We applied our leader detection algorithm to a real world network. Our results show that: (i) central hubs are usually incapable of having a strong effect in influencing the behavior of the entire network; (ii) there is a trade-off between how long the cascade chains are and how engaged each element of the chain is; (iii) to achieve maximum engagement it is better to target leaders in tightly connected communities, although for this last point we do not have conclusive evidence. We also included a case study in which we show how artists in different musical genres are spread through the network.

Many future developments are possible. The limited prominence that central hubs have on the overall network may be studied in conjunction with the problem of network controllability [212]. Alternative leader detection techniques, such as the ones presented in [213], can be confronted with our proposed algorithm. Finally, a deeper analysis of the properties of the Width, Depth and Strength measures can be performed, using additional techniques and exploiting data from other social media services like Twitter and Facebook.

Chapter 11

Conclusion

> Reasoning draws a conclusion, but does not make the conclusion certain, unless the mind discovers it by the path of experience.
>
> — *Roger Bacon*

In this thesis, we have highlighted how several network problem formulations need to be revised in order to comply with the inherent dynamics expressed by social phenomena. In order to better describe the role time plays in shaping the local structures as well as more complex topologies of social networks we organized our work in three parts.

Firstly, we identified a set of interesting problems which, in our opinion, can take advantage from the introduction of temporal dynamics both in their definition and in the analytical processes designed to solve them. For all these tasks we reported the classical definitions and analyzed the current state of art. Secondly, we presented our contribution to well-known problem of static network analysis discussing novel approaches able to address and solve both individual and collective issues. Finally, we reported our contribution to the emerging field of dynamic network analysis: moving from the results previously discussed, we described how to reformulate some static problem in order to involve time into the analytical process. In particular we propose time-aware approaches to solve Link Prediction and Community Discovery in evolving networks as well as a methodology to estimate social prominence in a musical taste diffusion scenario. The definition of scalable analytical instruments able to extract meaningful knowledge from complex evolving structures, i.e., social networks, will play a relevant role in the era of Big Data. Nowadays forecasting and nowcasting are among the most intriguing and complex issues to deal with: selectively tracking dynamic processes in real time will became in the next few years one of the main need for data owners. Our approach to the evolutionary community discovery problem, Tiles, has to be seen as one of the first attempts to deal online with structure perturbation generated by dynamic complex processes.

The future research directions for the analytical approaches proposed in thesis can be mainly divided in three tracks. The first line of research involves the temporal extension for those problems introduced in the 2^{nd} part of the thesis which were not addressed in the 3^{rd}. Network quantification and social engagement are certainly two very relevant case studies that can be fruitfully extended to capture the semantic dynamics of evolving social networks. Moreover, our approach to the link-based object ranking, Ubik, can be easily extended to comply with dynamic scenarios and to produce time-aware results that model expertise decay through time. A second track involves an in depth study and characterization of collective and individual temporal dynamics. Varying the temporal scale used to model dynamic processes different informations can be captured: restricting the frequency of observation to minutes or even seconds we are able to describe interaction trends and to study patterns of communication which correlate specific pair of individual; moving from

minutes to hours small sized frequent pattern begin to emerge describing well defined social ties; aggregating for days or even weeks, complex community structures start to appear; while observing a dynamic phenomenon for a longer period allow the identification of its own temporal "signature".
A third track of research involves the study of diffusive processes in dynamic network scenarios. Allowing the network topology to change over time while studying how a virus, an idea or even musical taste diffuse can lead to a better understanding of how and at which rate in a real world context viral phenomena manifest and generate cascades.
Our final goal will be to provide the scientific world with tools able to support the extraction of valuable knowledge from dynamic processes that shape network structures: indeed, we believe that nowadays network dynamics competencies should be part of the workbench of any complex system scientist.

Bibliography

[1] M. Magnani, A. Monreale, G. Rossetti, and F. Giannotti, "On multidimensional network measures," in *Italian Conference on Sistemi Evoluti per le Basi di Dati (SEBD)*, 2013.

[2] L. Pappalardo, G. Rossetti, and D. Pedreschi, "How well do we know each other? detecting tie strength in multidimensional social networks," in *ASONAM*, pp. 1040–1045, 2012.

[3] M. Coscia, G. Rossetti, D. Pennacchioli, D. Ceccarelli, and F. Giannotti, ""you know because i know": A multidimensional network approach to human resources problem," in *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), 2013*, pp. 434–441, 2013.

[4] M. Coscia, G. Rossetti, D. Pedreschi, and F. Giannotti, "Demon: a local-first discovery method for overlapping communities," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD*, 2012.

[5] M. Coscia, G. Rossetti, F. Giannotti, and D. Pedreschi, "Uncovering hierarchical and overlapping communities with a local-first approach," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2014.

[6] G. Rossetti, M. Berlingerio, and F. Giannotti, "Scalable link prediction on multidimensional networks," in *ICDM Workshops*, pp. 979–986, 2011.

[7] D. Pennacchioli, G. Rossetti, L. Pappalardo, D. Pedreschi, F. Giannotti, and M. Coscia, "The three dimensions of social prominence," in *Social Informatics*, 2013.

[8] G. Rossetti, "Tiles: Evolutionary community discovery in interaction networks." Temporal Networks in Human Dynamics Satellite Meeting European Conference on Complex Systems, 2014.

[9] R. Albert and A.-L. Barabási, "Statistical mechanics of complex networks," *Rev. Mod. Phys.*, vol. 74, no. 1, pp. 47–97, 2002.

[10] M. E. J. Newman, "The structure and function of complex networks," *SIAM Review*, vol. 45, no. 2, pp. 167–256, 2003.

[11] D. Watts and S. Strogatz, "Collective dynamics of small-world networks," *Nature*, no. 393, pp. 440–442, 1998.

[12] J. Leskovec, J. M. Kleinberg, and C. Faloutsos, "Graphs over time: densification laws, shrinking diameters and possible explanations," in *Proceedings of the 11th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD*, 2005.

[13] K. I. Goh, E. OH, H. Jeong, B. Kahng, and D. Kim, "Classification of scale-free networks," in *National Academy of Science*, 2002.

[14] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web," Technical Report 1999-66, Stanford InfoLab, November 1999.

[15] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, pp. 75–174, Feb. 2010.

[16] R. Solomonoff and A. Rapoport, "Connectivity of random nets," in *Bulletin of Math- ematical Biology*, pp. 107–117, 1951.

[17] P. Erdos and A. Renyi, "On random graphs," in *Publicationes Mathematicae*, 1959.

[18] S. Milgram, "The small world problem," in *Psychol*, 1967.

[19] J. Guare, "Six degrees of separation: A play," in *Vintage Books*, 1990.

[20] M. E. J. Newman and D. J. Watts, "Renormalization group analysis of the small-world network model," in *Phys. Letters*, 1999.

[21] S. H. Yook, H. Jeong, and A. L. Barabasi, "Modeling the internet's large-scale topology," in *Proceedings of the National Academy of Sciences*, 2002.

[22] R. Pastor-Satorras, A. Vazquez, and A. Vespignani, "Dynamical and correlation properties of the internet," in *Physical Review Letters*, 2001.

[23] L. A. Adamic, "The small world web conference," in *European Conference on Digital Libraries*, 1999.

[24] R. Albert, H. Jeong, and A. L. Barabasi, "Diameter of the world wide web," *Nature*, vol. 401, pp. 130–131, 1999.

[25] R. Muhamad, P. S. Dodds, and D. Watts, "An experimental study of search in global search networks," in *Science*, 2003.

[26] H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai, and A. L. Barabasi., "The large-scale organization of metabolic networks," *Nature*, 2000.

[27] A. L. Barabasi, H. Jeong, Z. Neda, E. Ravasz, A. Schubert, and T. Vicsek, "Evolution of the social network of scientific collaborations," *Physica*, 2002.

[28] M. E. J. Newman, "The structure of scientific collaboration networks," in *Proc. Natl. Acad. Sci.*, 2001.

[29] S. Horita, K. Oshio, Y. Osama, Y. Funabashi, K. Oka, and K. Kawamara, "Geometrical structure of the neuronal network of caenorhabditis elegans," *Physica*, 2001.

[30] J. Leskovec and E. Horvitz, "Worldwide buzz: Planetary-scale views on an instant-messaging network," tech. rep., Microsoft Research Technical Report MSR-TR- 2006-186, 2007.

[31] L. A. N. Amaral, A. Scala, M. Barthelemy, and H. E. Stanley, "Classes of behavior of small world networks," in *Natl. Acad. Sci.*, 2000.

[32] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On power-law relationships of the internet topology," in *SIGCOMM*, pp. 251–262, 1999.

[33] A. L. Barabási and R. Albert, "Emergence of scaling in random networks," *science*, vol. 286, no. 5439, pp. 509–512, 1999.

[34] J. M. Kleinberg, R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins, "The web as a graph: Measurements, models, and methods," in *International Conference on Combinatorics and Computing*, 1999.

[35] R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and E. Upfal, "Stochastic models for the web graph," in *41st IEEE Symp. on Foundations of Computer Science*, 2000.

[36] B. Goncalves, N. Perra, and A. Vespignani, "Modeling users' activity on twitter networks: Validation of dunbar's number," *PLoS One,*, vol. 6, 2011.

[37] R. I. M. Dunbar, "The social brain hypothesis," *Evol. Anthropol*, vol. 6, no. 5, 1998.

[38] J. Goldenberg and M. Levy, "Distance is not dead: Social interaction and geographical distance in the internet era," *CoRR abs/0906.3202*, 2009.

[39] D. Mok, B. Wellman, and J. A. Carrasco, "Does distance still matter in the age of the internet?," *Urban Studies*, vol. 45, 2009.

[40] J. P. Onnela, J. Saramäki, J. Hyvönen, G. Szabó, D. Lazer, K. Kaski, J. Kertész, and A. L. Barabási, "Structure and tie strengths in mobile communication networks," in *Proceedings of the National Academy of Sciences*, vol. 104, pp. 7332–7336, 2007.

[41] R. S. Burt, "Structural holes and good ideas," *American Journal of Sociology*, vol. 110, no. 2, 2004.

[42] M. Granovetter, "Getting a job: A study of contacts and careers," *University Of Chicago Press*, 1974.

[43] C. Schaefer and J. C. Coyne, "The health-related functions of social support," *Journal of Behavioral Medicine*, 1990.

[44] J. H. Fowler and N. A. Christakis, "Dynamic spread of happiness in a large social network: longitudinal analysis over 20 years in the framingham heart study," *Bmj Clinical Research Ed.*, vol. 337, no. 2, pp. a2338–a2338, 2008.

[45] D. Krackhardt and R. N. Stern, "Informal networks and organizational crises: An experimental simulation," *Social Psychology Quarterly*, vol. 51, no. 2, pp. 123–140, 1988.

[46] M. Granovetter, "The strength of weak ties," *American journal of sociology*, vol. 78, no. 6, p. l, 1973.

[47] J. M. Kleinberg, "Authoritative sources in a hyperlinked environment," *J. ACM*, vol. 46, pp. 604–632, Sept. 1999.

[48] A. Esuli and F. Sebastiani, "Sentiment quantification," *IEEE Intelligent Systems*, vol. 25, no. 4, pp. 72–75, 2010.

[49] G. Forman, "Quantifying trends accurately despite classifier error and class imbalance," in *In KDD 2006*, pp. 157–166.

[50] G. Forman, "Quantifying counts and costs via classification," *DMKD*, vol. 17, no. 2, pp. 164–206, 2008.

[51] L. Tang, H. Gao, and H. Liu, "Network quantification despite biased labels," in *MLG 2010*, pp. 147–154.

[52] J. Xue and G. Weiss, "Quantification and semi-supervised classification methods for handling changes in class distribution," in *KDD 2009*, pp. 897–906.

[53] N. Lin and W. M. Ensel, "Social resources and strength of ties: Structural factors in occupational status attainment," *American Sociological Review*, vol. 46, no. 4, 1981.

[54] B. Wellman and S. Wortley, "Different strokes from different folks: Community ties and social support," *The American Journal of Sociology*, vol. 96, no. 3, 1990.

[55] R. Burt, "Structural holes: The social structure of competition," *Harvard University Press*, 1995.

[56] P. V. Marsden and K. E. Campbell, "Measuring tie strength," *Social Forces*, vol. 63, no. 2, 1990.

[57] E. Gilbert and K. Karahalios, "Predicting tie strength with social media," in *Proceedings of the 27th international conference on Human factors in computing systems*, 2009.

[58] M. Szell, R. Lambiotte, and S. Thurner, "Trade, conflict and sentiments: Multi-relational organization of large-scale social networks," *arXiv.org, 1003.5137*, 2010.

[59] L. Lü and T. Zhou, "Link prediction in complex networks: A survey," *Physica A: Statistical Mechanics and its Applications*, vol. 390, no. 6, pp. 1150–1170, 2011.

[60] D. M. Pennock, G. W. Flake, S. Lawrence, E. J. Glover, and C. L. Giles, "Winners don't take all: Characterizing the competition for links on the web," *PNAS*, vol. 99, pp. 5207–5211, Apr. 2002.

[61] L. A. Adamic and E. Adar, "Friends and neighbors on the web," *Social networks*, vol. 25, no. 3, pp. 211–230, 2003.

[62] M. E. J. Newman, "Clustering and preferential attachment in growing networks," *PHYS.REV.E*, vol. 64, p. 025102, 2001.

[63] L. Nowell and J. Kleinberg, "The link prediction problem for social networks," in *CIKM '03 Proceedings of the twelfth international conference on Information and knowledge management*, 2003.

[64] A. Clauset, C. Moore, and M. Newman, "Hierarchical structure and the prediction of missing links in networks," *Nature*, 2008.

[65] C. Zhou, L. Zemanová, G. Zamora, C. Hilgetag, and J. Kurths, "Hierarchical organization unveiled by functional connectivity in complex brain networks," *Phys. Rev. Lett.*, 2006.

[66] S. Redner, "Teasing out the missing links," *Nature*, 2008.

[67] P. H. M. Huss, "Currency and commodity metabolites: their identification and relation to the modularity of metabolic networks," *IET Syst. Biol.*, 2007.

[68] N. Friedman, L. Getoor, D. Koller, and A. Pfeffer, "Learning probabilistic relational models," in *16th International Joint Conference on Artificial Intelligence*, 1999.

[69] B. Taskar, M. F. Wong, P. Abbeel, and D. Koller, "Link prediction in relational data," in *Neural Information Precessing Systems*, 2004.

[70] D. Heckerman, C. Meek, and D. Koller, "Probabilistic entity-relationship models, prms, and plate models," in *21st International Conference on Machine Learning*, 2004.

[71] C. A. Bliss, M. R. Frank, C. M. Danforth, and P. S. Dodds, "An evolutionary algorithm approach to link prediction in dynamic social networks," *arXiv preprint arXiv:1304.6257*, 2013.

[72] Z. Bao, Y. Zeng, and Y. Tay, "sonlp: Social network link prediction by principal component regression," in *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 364–371, ACM, 2013.

[73] B. Bringmann, M. Berlingerio, F. Bonchi, and A. Gionis, "Learning and predicting the evolution of social networks," *IEEE Intelligent Systems*, vol. 25, no. 4, pp. 26–35, 2010.

[74] M. Bilgic, G. M. Namata, and L. Getoor, "Combining collective classification and link prediction," ICDMW '07, (Washington, DC, USA), pp. 381–386, IEEE Computer Society, 2007.

[75] M. Pujari and R. Kanawati, "Supervised rank aggregation approach for link prediction in complex networks," in *Proceedings of the 21st international conference companion on World Wide Web*, pp. 1189–1196, ACM, 2012.

[76] N. Shibata, Y. Kajikawa, and I. Sakata, "Link prediction in citation networks," *Journal of the American Society for Information Science and Technology*, vol. 63, no. 1, pp. 78–85, 2012.

[77] S. Spiegel, J. Clausen, S. Albayrak, and J. Kunegis, "Link prediction on evolving data using tensor factorization," in *New Frontiers in Applied Data Mining*, pp. 100–110, Springer, 2012.

[78] R. N. Lichtenwalter, J. T. Lussier, and N. V. Chawla, "New perspectives and methods in link prediction," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 243–252, ACM, 2010.

[79] S. Soundarajan and J. Hopcroft, "Using community information to improve the precision of link prediction methods," in *Proceedings of the 21st international conference companion on World Wide Web*, pp. 607–608, ACM, 2012.

[80] P. Sarkar, D. Chakrabarti, and M. Jordan, "Nonparametric link prediction in dynamic networks," *arXiv preprint arXiv:1206.6394*, 2012.

[81] P. R. da Silva Soares and R. Bastos Cavalcante Prudencio, "Time series based link prediction," in *Neural Networks (IJCNN), The 2012 International Joint Conference on*, pp. 1–7, IEEE, 2012.

[82] X. Feng, J. Zhao, and K. Xu, "Link prediction in complex networks: a clustering perspective," *The European Physical Journal B*, vol. 85, no. 1, pp. 1–9, 2012.

[83] K. Jahanbakhsh, V. King, and G. C. Shoja, "Predicting human contacts in mobile social networks using supervised learning," in *Proceedings of the Fourth Annual Workshop on Simplifying Complex Networks for Practitioners*, pp. 37–42, ACM, 2012.

[84] M. Fire, R. Puzis, and Y. Elovici, "Link prediction in highly fractional data sets," in *Handbook of Computational Approaches to Counterterrorism*, pp. 283–300, Springer, 2013.

[85] Y. Xu and D. Rockmore, "Feature selection for link prediction," in *Proceedings of the 5th Ph. D. workshop on Information and knowledge*, pp. 25–32, ACM, 2012.

[86] R. Lichtnwalter and N. V. Chawla, "Link prediction: fair and effective evaluation," in *Advances in Social Networks Analysis and Mining (ASONAM), 2012 IEEE/ACM International Conference on*, pp. 376–383, IEEE, 2012.

[87] A. Veloso, M. A. Gonçalves, and W. M. Jr., "Competence-conscious associative rank aggregation," *JIDM*, vol. 2, no. 3, pp. 337–352, 2011.

[88] R. Lempel and S. Moran, "The stochastic approach for link-structure analysis (SALSA) and the TKC effect," *Computer Networks (Amsterdam, Netherlands: 1999)*, vol. 33, no. 1–6, pp. 387–401, 2000.

[89] A. Sidiropoulos and Y. Manolopoulos, "Generalized comparison of graph-based ranking algorithms for publications and authors," *J. Syst. Softw.*, vol. 79, no. 12, 2006.

[90] T. H. Haveliwala, "Topic-sensitive PageRank: A context-sensitive ranking algorithm for Web search," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 4, pp. 784–796, 2003.

[91] M. K.-P. Ng, X. Li, and Y. Ye, "Multirank: co-ranking for objects and relations in multi-relational data," in *SIGKDD*, (New York, NY, USA), pp. 1217–1225, ACM, 2011.

[92] X. Li, M. K. Ng, and Y. Ye, "Har: Hub, authority and relevance scores in multi-relational data for query search," in *SDM*, pp. 141–152, 2012.

[93] J. Tang, J. Zhang, R. Jin, Z. Yang, K. Cai, L. Zhang, and Z. Su, "Topic level expertise search over heterogeneous networks," *Machine Learning*, vol. 82, no. 2, pp. 211–237, 2011.

[94] T. G. Kolda, B. W. Bader, and J. P. Kenny, "Higher-Order web link analysis using multilinear algebra," in *ICDM*, (Washington, DC, USA), pp. 242–249, IEEE Computer Society, 2005.

[95] M. Berlingerio, M. Coscia, F. Giannotti, A. Monreale, and D. Pedreschi, "Multidimensional networks: foundations of structural analysis," *World Wide Web Journal*, pp. 1–27, 2012.

[96] P. J. Mucha, T. Richardson, K. Macon, M. A. Porter, and J. Onnela, "Community structure in time-dependent, multiscale, and multiplex networks," *Science 328, 876*, 2010.

[97] M. Berlingerio, M. Coscia, F. Giannotti, A. Monreale, and D. Pedreschi, "The pursuit of hubbiness: Analysis of hubs in large multidimensional networks," *Journal of Computer Science*, vol. 2, no. 3, pp. 223–237, 2011.

[98] M. Kivelä, A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno, and M. A. Porter, "Multilayer networks," *arXiv:1309.7233 [physics.soc-ph]*, 2013.

[99] S. A. Catanese, P. D. Meo, E. Ferrara, G. Fiumara, and A. Provetti, "Crawling facebook for social network analysis purposes," in *WIMS*, 2011.

[100] J. Iturrioz, O. Diaz, and C. Arellano, "Towards federated web2.0 sites: the tagmas approach," in *International Workshop on Tagging and Metadata for Social Information Organization*, 2007.

[101] M. Szomszor, I. Cantador, , and H. Alani, "Correlating user profiles from multiple folksonomies," in *HT*, 2008.

[102] F. Abel, N. Henze, E. Herder, and D. Krause, "Interweaving public user profiles on the web," in *UMAP*, 2010.

[103] A. Stewart, E. Diaz-Aviles, W. Nejdl, L. B. Marinho, A. Nanopoulos, and L. Schmidt-Thieme, "Cross-tagging for personalized open social networking," in *HT*, 2009.

[104] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," Nov. 2012.

[105] M. Coscia, F. Giannotti, and D. Pedreschi, "A classification for community discovery methods in complex networks," *Statistical Analysis and Data Mining*, 2011.

[106] S. Papadimitriou, J. Sun, C. Faloutsos, and P. S. Yu, "Hierarchical, parameter-free community discovery," in *ECML PKDD*, pp. 170–187, 2008.

[107] A. Clauset, M. E. J. Newman, and C. Moore, "Finding community structure in very large networks," *Physical Review E*, vol. 70, p. 066111, 2004.

[108] M. E. J. Newman, "Modularity and community structure in networks," *Proceedings of the National Academy of Sciences*, vol. 103, pp. 8577–8582, June 2006.

[109] S. Fortunato and M. Barthélemy, "Resolution limit in community detection," *PNAS*, vol. 104, pp. 36–41, Jan. 2007.

[110] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, p. P10008, 2008.

[111] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences*, vol. 99, pp. 7821–7826, June 2002.

[112] J. P. Bagrow and E. M. Bollt, "Local method for detecting communities," *Physical Review E*, vol. 72, pp. 046108+, Oct. 2005.

[113] A. Lancichinetti and S. Fortunato, "Community detection algorithms: A comparative analysis," *Physical Review E*, vol. 80, pp. 056117–+, Nov. 2009.

[114] P. Pons and M. Latapy, "Computing communities in large networks using random walks," *J. Graph Algorithms Appl.*, vol. 10, no. 2, pp. 191–218, 2006.

[115] Y.-Y. Ahn, J. P. Bagrow, and S. Lehmann, "Link communities reveal multiscale complexity in networks," *Nature*, vol. 466, pp. 761–764, June 2010.

[116] T. S. Evans and R. Lambiotte, "Line graphs, link partitions, and overlapping communities," *Physical Review E*, vol. 80, pp. 016105+, July 2009.

[117] I. Derényi, G. Palla, and T. Vicsek, "Clique Percolation in Random Networks," *Physical Review Letters*, vol. 94, pp. 160202+, Apr. 2005.

[118] K. Henderson, T. Eliassi-Rad, S. Papadimitriou, and C. Faloutsos, "Hcdf: A hybrid community discovery framework," in *SDM*, pp. 754–765, 2010.

[119] J. J. McAuley and J. Leskovec, "Learning to discover social circles in ego networks.," in *NIPS* (P. L. Bartlett, F. C. N. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds.), pp. 548–556, 2012.

[120] U. N. Raghavan, R. A., and S. Kumara, "Near linear time algorithm to detect community structures in large-scale networks," *Physical Review*, 2007.

[121] M. V. Nguyen, "Community evolution in a scientific collaboration network," *CEC IEEE*, 2012.

[122] M. Goldberg, M. Magdon-Ismail, S. Nambirajan, and J. Thompson, "Tracking and predicting evolution of social communities," *IEEE Third Int Conference on Privacy, Security, Risk and Trust and Third Int Conference on Social Computing*, 2011.

[123] M. Takaffoli, F. Sangi, J. Fagnan, and O. Zaïane, "Modec-modeling and detecting evolutions of communities," *ICWSM*, 2011.

[124] G. Palla, A. Barabási, and T. Vicsek, "Quantifying social group evolution," *Nature*, 2007.

[125] C. Tantipathananandh, T. Berger-Wolf, and D. Kempe, "A framework for community identification in dynamic social networks," *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD*, 2007.

[126] D. Chakrabarti, R. Kumar, and A. Tomkins, "Evolutionary clustering," *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD*, 2006.

[127] A. Sitaram, P. Srinivasan, and U. Duygu, "An event-based framework for characterizing the evolutionary behavior of interaction graphs," *ACM Transactions on Knowledge Discovery from Data*, 2009.

[128] Y. Sun, J. Tang, J. Han, M. Gupta, and B. Zhao, "Community evolution detection in dynamic heterogeneous information networks," *Proceedings of the Eighth Workshop on Mining and Learning with Graphs - MLG*, 2010.

[129] J. Xie, M. Chen, and B. K. Szymanski, "Labelrankt: Incremental community detection in dynamic networks via label propagation," *CoRR*, vol. abs/1305.2006, 2013.

[130] G. Qi, C. C. Aggarwal, and T. S. Huang, "Online community detection in social sensing," *Proceedings of the sixth ACM international conference on Web search and data mining - WSDM*, 2013.

[131] Y. Lin, Y. Chi, and S. Zhu, "Facetnet: a framework for analyzing communities and their evolutions in dynamic networks," *Proceedings of the 17th international conference on World Wide Web*, 2008.

[132] R. Cazabet, F. Amblard, and C. Hanachi, "Detection of overlapping communities in dynamical social networks," in *SocialCom*, pp. 309–314, 2010.

[133] Q. Zhao, S. S. Bhowmick, X. Zheng, and K. Yi, "Characterizing and predicting community members from evolutionary and heterogeneous networks," *Proceeding of the 17th ACM conference on Information and knowledge mining - CIKM*, 2008.

[134] D. D. Lewis, "Evaluating and optmizing autonomous text classification systems," in *In SIGIR 1995*, pp. 246–254.

[135] G. Forman, "Counting positives accurately despite inaccurate classification," in *In ECML*, pp. 564–575, 2005.

[136] A. Bella, C. Ferri, J. Hernández-Orallo, and M. J. Ramírez-Quintana, "Quantification via probability estimators," in *ICDM 2010*, pp. 737–742.

[137] A. Esuli and F. Sebastiani, "Machines that learn how to code open-ended survey data," *International Journal of Market Research*, vol. 52, no. 6, pp. 775–800, 2010.

[138] D. J. Hopkins and G. King, "A method of automated nonparametric content analysis for social science," *American Journal of Political Science*, vol. 54, no. 1, pp. 229–247, 2010.

[139] L. A. Goodman, "Snowball sampling," *The Annals of Mathematical Statistics*, vol. 32, no. 1, pp. 148–170, 1961.

[140] M. Spreen and R. Zwaagstra, "Personal network sampling, outdegree analysis and multilevel analysis: Introducing the network concept in studies of hidden populations," *International Sociology*, vol. 9, no. 4, pp. 475–491, 1994.

[141] E. Deaux and J. W. Callaghan, "Key informant versus self-report estimates of health-risk behavior," *Evaluation Review*, vol. 9, no. 3, p. 365, 1985.

[142] J. K. Watters and P. Biernacki, "Targeted sampling: options for the study of hidden populations," *Social Problems*, pp. 416–430, 1989.

[143] D. D. Heckathorn, "Respondent-driven sampling: a new approach to the study of hidden populations," *Social problems*, pp. 174–199, 1997.

[144] L. Milli, A. Monreale, G. Rossetti, F. Giannotti, D. Pedreschi, and F. Sebastiani, "Quantification trees," *2013 IEEE 13th International Conference on Data Mining*, vol. 0, pp. 528–536, 2013.

[145] Y. Zhu, E. Zhong, S. J. Pan, X. Wang, M. Zhou, and Q. Y. 0001, "Predicting user activity level in social networks.," in *CIKM* (Q. He, A. Iyengar, W. Nejdl, J. Pei, and R. Rastogi, eds.), pp. 159–168, ACM, 2013.

[146] Y. Richter, E. Yom-Tov, and N. Slonim, "Predicting customer churn in mobile networks through analysis of social groups," in *SDM*, pp. 732–741, 2010.

[147] R. J. Oentaryo, E.-P. Lim, D. Lo, F. Zhu, and P. K. Prasetyo, "Collective churn prediction in social network.," in *ASONAM*, pp. 210–214, IEEE Computer Society, 2012.

[148] R. Bhatt, V. Chaoji, and R. Parekh, "Predicting product adoption in large-scale social networks.," in *CIKM* (J. Huang, N. Koudas, G. J. F. Jones, X. Wu, K. Collins-Thompson, and A. An, eds.), pp. 1039–1048, ACM, 2010.

[149] P. Domingos and M. Richardson, "Mining the network value of customers," in *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 57–66, ACM Press, 2001.

[150] J. D. Hartline, V. S. Mirrokni, and M. Sundararajan, "Optimal marketing strategies over social networks.," in *WWW* (J. Huai, R. Chen, H.-W. Hon, Y. Liu, W.-Y. Ma, A. Tomkins, and X. Z. 0001, eds.), pp. 189–198, ACM, 2008.

[151] A. Bagherjeiran and R. Parekh, "Combining behavioral and social network data for online advertising.," in *ICDM Workshops*, pp. 837–846, IEEE Computer Society, 2008.

[152] V. Colizza, A. Barrat, M. Barthelemy, A. Valleron, and A. Vespignani, "Modeling the worldwide spread of pandemic influenza: Baseline case and containment interventions," *PLoS Medicine*, 2007.

[153] P. W. and. M. C Gonzàlez, C. A. Hidalgo, and A. Barabási, "Understanding the spreading patterns of mobile phone viruses," *Science*, 2009.

[154] R. S. Burt, "Social contagion and innovation: Cohesion versus structural equivalence," *American Journal of Sociology*, 1987.

[155] M. Coscia, "Competition and success in the meme pool: a case study on quickmeme.com," *ICWSM*, 2013.

[156] W. O. Kermack and A. G. McKendrick, "A contribution to the mathematical theory of epidemics," *The Royal Society of London Series A*, vol. 115, no. 772, pp. 700–721, 1927.

[157] R. Pastor-Satorras and A. Vespignani, "Epidemic spreading in scale-free networks," *Physical review letters*, vol. 86, no. 14, pp. 3200–3203, 2001.

[158] N. A. Christakis and J. H. Fowler, "The spread of obesity in a large social network over 32 years.," *New England Journal of Medicine*, vol. 357, no. 4, pp. 370–379, 2007.

[159] N. A. Christakis and J. H. Fowler, "The collective dynamics of smoking in a large social network.," *New England Jou. of Medicine*, vol. 358, no. 21, pp. 2249–2258, 2008.

[160] E. M. Rogers, *Diffusion of Innovation*. Free Press, 5th ed., 2003.

[161] P. Holme and J. Saramäki, *Temporal Networks*. Springer, 2013.

[162] V. Nicosia, J. Tang, C. Mascolo, M. Musolesi, G. Russo, and V. Latora, "Graph metrics for temporal networks," *CoRR*, 2013.

[163] B. Min and K.-I. Goh, *Burstiness: Measures, Models, and Dynamic Consequences*. Springer, 2013.

[164] R. S. Caceres, T. Y. Berger-Wolf, and R. Grossman, "Temporal scale of processes in dynamic networks," in *ICDM Workshops*, 201.

[165] A. V. Mantzaris and D. Higham, *Inferring and Calibrating Triadic Closure in a Dynamic Network*. Springer, 2013.

[166] L. Kovanen, K. Kaski, J. Kertész, and J. Saramäki, "Temporal motifs reveal homophily, gender-specific patterns and group talk in mobile communication networks," *CoRR*, 2013.

[167] L. Kovanen, M. Karsai, K. Kaski, J. Kertész, and J. Saramäki, "Temporal motifs in time-dependent networks," *CoRR*, 2011.

[168] F. Karimi and P. Holme, *A Temporal Network Version of Watts's Cascade Model*. Springer, 2013.

[169] N. Perra, B. Gonc alves, R. Pastor-Satorras, and A. Vespignani, "Activity driven modeling of time varying networks," in *Sci. Rep.*, 2012.

[170] L. Rocha, A. Decuyper, and V. Blondel, "Epidemics on a stochastic model of temporal network," in *arXiv:1204.5421*, 2012.

[171] M. Magnani and L. Rossi, "Pareto distance for multi-layer network analysis," in *International Conference on Social Computing, Behavioral Modeling and Prediction*, Springer, Berlin, 2013.

[172] M. Magnani and L. Rossi, "The ml-model for multi layer network analysis," in *IEEE International conference on Advances in Social Network Analysis and Mining*, IEEE Computer Society, Los Alamitos, 2011. isbn: 978-0-769-54375-8.

[173] M. Magnani, B. Micenková, and L. Rossi, "Combinatorial analysis of multiple networks," 2013.

[174] C. Haythornthwaite and B. Wellman, "Work, friendship, and media use for information exchange in a networked organization," *J. Am. Soc. Inf. Sci.*, vol. 49, no. 12, pp. 1101–1114, 1998.

[175] C. Haythornthwaite, "Strong, weak, and latent ties and the impact of new media," *Information Society*, 2002.

[176] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: Homophily in social networks," *Annual Review of Sociology*, vol. 27, no. 1, pp. 415–444, 2001.

[177] E. Moretti, "Estimating the social return to higher education: evidence from longitudinal and repeated cross-sectional data," *Journal of Econometrics*, vol. 121, no. 1-2, pp. 175–212, 2004.

[178] M. F. Porter, "An Algorithm for Suffix Stripping," *Program*, vol. 14, no. 3, pp. 130–137, 1980.

[179] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1 ed., Feb. 1993.

[180] S. Fortunato, M. Boguna, A. Flammini, and F. Menczer, "Approximating pagerank from in-degree," in *Lecture Notes in Computer Science*, pp. 59–71, 2008.

[181] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *OSDI*, pp. 137–150, 2004.

[182] A. Lancichinetti and S. Fortunato, "Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities," *Phys. Rev. E*, vol. 80, p. 016118, July 2009.

[183] J. Xie and B. Szymanski, "Towards linear time overlapping community detection in social networks," *PAKDD*, 2012.

[184] J. Xie, B. K. Szymanski, and X. Liu, "Slpa: Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process," in *ICDM Workshops*, pp. 344–349, 2011.

[185] Y. Jaewon and J. Leskovec, "Overlapping community detection at scale: A nonnegative matrix factorization approach," in *WSDM*, 2013.

[186] G. Tsoumakas and I. Katakis, "Multi label classification: An overview," *International Journal of Data Warehousing and Mining*, vol. 3, no. 3, pp. 1–13, 2007.

[187] S. Godbole and S. Sarawagi, "Discriminative methods for multi-labeled classification," in *PAKDD*, pp. 22–30, 2004.

[188] A. Lancichinetti, S. Fortunato, and J. Kertész, "Detecting the overlapping and hierarchical community structure in complex networks," *New Journal of Physics*, vol. 11, no. 3, p. 033015, 2009.

[189] A. F. McDaid, D. Greene, and N. Hurley, "Normalized mutual information to evaluate overlapping community finding algorithms," Oct. 2011.

[190] M. Berlingerio, M. Coscia, and F. Giannotti, "Finding and characterizing communities in multidimensional networks," in *ASONAM*, pp. 490–494, 2011.

[191] A. S. Sinan Aral, Lev Muchnika, "Distinguishing influence-based contagion from homophily-driven diffusion in dynamic networks," 2009.

[192] C. R. Shalizi and A. C. Thomas, "Homophily and contagion are generically confounded in observational social network studies," Tech. Rep. arXiv:1004.4704, Apr 2010. Comments: 24 pages, 9 figures.

[193] N. Z. Gong, W. Xu, L. Huang, P. Mittal, E. Stefanov, V. Sekar, and D. Song, "Evolution of social-attribute networks: Measurements, modeling, and implications using google+," *CoRR*, vol. abs/1209.0835, 2012.

[194] M. Rosvall and C. T. Bergstrom, "Multilevel compression of random walks on networks reveals hierarchical organization in large integrated systems," *PloS one*, vol. 6, no. 4, p. e18209, 2011.

[195] S. A. Macskassy and F. Provost, "Classification in networked data: A toolkit and a univariate case study," *J. Mach. Learn. Res.*, vol. 8, pp. 935–983, Dec. 2007.

[196] M. E. J. Newman, "Mixing patterns in networks," *Phys. Rev. E*, vol. 67, p. 026126, 2003.

[197] Y. Tsuruoka, J. Tsujii, and S. Ananiadou, "Stochastic gradient descent training for l1-regularized log-linear models with cumulative penalty.," in *ACL/IJCNLP* (K.-Y. Su, J. Su, and J. Wiebe, eds.), pp. 477–485, The Association for Computer Linguistics, 2009.

[198] T. Zhang, "Solving large scale linear prediction problems using stochastic gradient descent algorithms.," in *ICML* (C. E. Brodley, ed.), vol. 69 of *ACM International Conference Proceeding Series*, ACM, 2004.

[199] J. Davis and M. Goadrich, "The relationship between precision-recall and roc curves," in *23rd International Conference on Machine Learning*, 2006.

[200] D. Wang, D. Pedreschi, C. Song, F. Giannotti, and A. L. Barabasi, "Human mobility, social ties, and link prediction," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2011.

[201] M. E. Newman, "Clustering and preferential attachment in growing networks," *Physical Review E*, vol. 64, no. 2, p. 025102, 2001.

[202] G. Salton and M. J. McGill, "Introduction to modern information retrieval," 1983.

[203] C. W.-k. Leung, E.-P. Lim, D. Lo, and J. Weng, "Mining interesting link formation rules in social networks," CIKM '10, (New York, USA), pp. 209–218, ACM, 2010.

[204] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *Nature*, vol. 435, pp. 814–818, June 2005.

[205] B. Viswanath, A. Mislove, M. Cha, and P. K. Gummadi, "On the evolution of user interaction in facebook," *WOSN*, 2009.

[206] M. Cha, H. Haddadi, F. Benevenuto, and K. Gummadi, "Measuring user influence in twitter: The million follower fallacy," in *ICWSM*, 2010.

[207] M. Berlingerio, M. Coscia, and F. Giannotti, "Mining the temporal dimension of the information propagation," in *IDA*, pp. 237–248, 2009.

[208] T. Kohonen, "The self-organizing map," *IEEE*, vol. 78, pp. 1464–1480, 1990.

[209] U. Kumar and Y. Dhamija, "A comparative analysis of som neural network with k-means clustering algorithm," *Proceedings of IEEE International Conference on Management of Innovation and Technology*, pp. 55–59, 2004.

[210] M. Y. Kiang and A. Kumar, "A comparative analysis of an extended som network and k-means analysis," *Int. J. Know.-Based Intell. Eng. Syst.*, vol. 8, no. 1, pp. 9–15, 2004.

[211] L. P. Cordella, P. Foggia, C. Sansone, and M. Vento, "A (sub)graph isomorphism algorithm for matching large graphs," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, no. 10, pp. 1367–1372, 2004.

[212] Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabasi, "Controllability of complex networks," *Nature*, vol. 473, pp. 167–173, May 2011.

[213] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan, "Discovering leaders from community actions," in *CIKM*, pp. 499–508, 2008.

Index