Politecnico di Milano
Master Degree in Data Engineering
Dipartimento di Elettronica, Informazione e Bioingegneria

# Community Analysis Using Graph Representation Learning On Social Networks

Advisor: Prof. Marco Brambilla

Master Thesis of:
Mattia Gasparini
Matr. 853965

Academic Year 2016-2017

*"The bad news is time flies.*
*The good news is you're the pilot."*

Michael Altshuler

# Acknowledgements

Desidero, innanzitutto, ringraziare la mia famiglia, che mi ha permesso di raggiungere questo obiettivo, supportandomi in tutte le situazioni e non facendomi mancare mai nulla.

Un grazie di cuore alla mia amata Federica, per essermi stata sempre vicina, avermi sopportato in questo lungo percorso e reso migliore ogni mia giornata.

Voglio anche esprimere la mia gratitudine nei confronti del professor Marco Brambilla, che mi ha introdotto e fatto appassionare alla *data science*, prima con l'indimenticabile esperienza ad Harvard e poi con questo lavoro di tesi.

Infine, voglio ringraziare tutti i miei amici e i miei compagni di università, senza i quali la vita scolastica e quella di tutti i giorni sarebbe davvero insostenibile: grazie anche a voi per non avermi abbandonato e aver reso fantastici questi anni trascorsi assieme.

# Abstract

In a world more and more connected, there is the opportunity to model this extreme degree of relationship among people in order to discover new and more complex patterns. A *graph* is the mathematical model that can be better exploited in the Web 2.0 era: the era of social networks. In fact, it is the model that perfectly fits for representing the interactions on platforms such as Facebook or Instagram: users can interact in many different ways, creating posts, putting "likes" and mentioning other users and in this way they incrementally build an enormous graph. This graph becomes much bigger if a set of interacting users, a *community*, is considered together.

This work deals with the problem of exploiting the network structure in order to map similarities between users inside communities detected on on-line social networks. The objective is the definition of a method to handle in an efficient way the heterogeneity of a social network, in order to encode all the data needed in a simpler graph model. The method presented allows to extract a "classical" social network, with only user nodes, from a much more complex network, without losing the necessary information to effectively capture users behaviour. Using this approach, two weighted, homogeneous networks are defined: the hashtags network and the mentions network. For the first network, the weights are given by the number of common hashtags used by each pair of users, while for the second they are the number of mentions made by each user. A very recent technique, known as *representation learning*, is then applied to these networks in order to describe user nodes in term of a continuous feature vector, that is used to perform classification and clustering.

In the first experiment, the resulting graphs are used to generate features that give a description as rich as possible of the users inside the network. These features are combined to train the model for a classification task that tries to discriminate between "consumer" and "non-consumer" users. Since the networks reduction allows to minimize the number of nodes, it is also possible to evaluate the influence of a broader set of users inside the same classification task: in fact, not labelled users contribute to the definition of relationship with the labelled ones, increasing the descriptive information of the features. In all the tests performed, the baseline, defined using features extracted from the account, is always overcome. In the second experiment, a similar process is developed using an unsupervised method. The objective is to discover sub-communities inside the principal ones, extending the classical problem of *community detection*. The set of features extracted are used as input for the K-means algorithm and the output defines a set of sub-communities that are validated with the help of domain experts. Their feedback, combined with a set of labels extracted from the networks, shows that the users can be divided in meaningful categories, thus verifying the power of the method in discovery hidden patterns.

# Sommario

In un mondo sempre più connesso, vi è l'opportunità di modellizzare l'estremo grado di relazione tra le persone in modo da scoprire nuovi e più complessi patterns. Un *grafo* è il modello matematico che può essere meglio sfruttato nell'era del Web 2.0: l'era dei social network. Infatti, è il modello che si adatta perfettamente a rappresentare le interazioni su piattaforme come Facebook o Instagram: gli utenti possono interagire in molti modi diversi, creando post, mettendo "like" e menzionando altri utenti e in questo modo costruiscono incrementalmente un enorme grafo. Questo grafo diventa molto più grande se un gruppo di utenti che interagiscono tra loro, una *community*, viene considerato insieme.

Questo lavoro affronta il problema di sfruttare la struttura a rete in modo da mappare similarità tra utenti all'interno di community individuate su social network on-line. L'obiettivo è la definizione di un metodo per gestire in modo efficiente l'eterogeneità intrinseca di un social network, in modo da codificare tutti i dati necessari in un modello a grafo più semplice. Il metodo presentato permette di estrarre una rete sociale "classica", contenente solo nodi di tipo utente, da una rete molto più complessa, senza perdere le informazioni necessarie per catturare efficacemente il comportamento degli utenti. Usando questo approccio, due reti pesate e omogenee sono definite: la rete degli hashtag e quella delle menzioni. Per la prima rete, i pesi sono ottenuti dal numero di hashtag in comune per ogni coppia di utenti, mentre per la seconda sono dati dal numero di menzioni fatte da ogni utente. Una tecnica molto recente, conosciuta come *representation learning*, è quindi applicata a queste reti in modo da descrivere i nodi utente come un vettore di feature continue, che è utilizzato per svolgere classificazione e clustering.

Nel primo esperimento svolto, i grafi risultanti sono utilizzati per generare feature in modo da avere una descrizione più ricca possibile degli utenti all'interno della rete. Queste feature sono combinate per svolgere il training di un modello in modo da risolvere un task di classificazione: la discriminazione tra utenti "consumer" e "non consumer". Poichè la riduzione delle reti permette di minimizzare il numero di nodi, è anche possibile valutare l'influenza di un gruppo di utenti più ampio all'interno dello stesso task di classificazione: infatti, gli utenti non classificati contribuiscono alla definizione di relazioni con quelli classificati, incrementando le informazioni descrittive delle feature. In tutti i test svolti, la baseline, definita utilizzando feature estratte dagli account, è sempre superata. Nel secondo esperimento, un processo similare viene sviluppato usando un metodo non supervisionato. L'obiettivo è scoprire sotto-community all'interno di quelle principali, estendendo il problema classico noto come *community detection*. Le feature estratte sono utilizzate come input per l'algoritmo K-means e l'output definisce un insieme di sotto-community che sono validate con l'aiuto di esperti di dominio. Il loro feedback, combinato con un insieme di label estratte dalle reti, mostra che gli utenti possono essere suddivisi in categorie significative, verificando quindi la potenza del metodo nello scoprire pattern nascosti.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Context and Problem Statement

Graphs can be used to model different domains: the first usage dates back to 1736, when the mathematician Leonhard Euler used graphs to solve the ”Seven Bridges of Koningsberg Problem”, proving that it had no solution and laid the foundation of graph theory. Since then, graphs have become of great importance in many fields: transportation and distribution problems, network design, production planning, resource management, sociology, biology. For the purpose of this work, the focus is on sociology and in particular on *social networks*. In fact, a graph is the model that perfectly fits for representing the interactions on platforms such as Facebook or Instagram: users interact in many different ways, creating posts, putting ”likes” and mentioning other users, building incrementally an enormous graph.
Social Network Analysis (SNA) is the process of investigating social structures through the use of networks and graph theory [27]. It has its theoretical roots in the work of early sociologists such as Georg Simmel and Émile Durkheim, who wrote about the importance of studying patterns of relationships that connect social actors. Social scientists have used the concept of ”social networks” since early in the 20th century to connote complex sets of relationships between members of social systems at all scales, from interpersonal to international. The researches present in literature cover a very high number of topics, with the common denominator of a quite simple network structure: nodes represent users and edges represent a specific interaction between them, which can be messages exchanged on an on-line college platform [4], chat on on-line games [28] or phone calls made inside the country [3]. Nowadays, the concept of social network is completely disrupted: inside social media domain, users relate with each other in forms that where impossible to think before: one can be interested in all the content produced by another user and so, ”following” its profile, he does not miss anything, another one wants to explicitly show a content to a friend and so it can ”mention” the other in the post or directly ”retweet” or ”re-pin” the content, one can appreciate a content putting a like and share its opinion with a comment. This set of possibilities generates networks much more complex, in which the standard techniques cannot be easily applied.

New techniques and approaches are needed to handle this complexity: representation learning is a very recent field, that allows to encode the nodes of a graph inside a continuous feature vector, that can be used to perform machine learning tasks. In this way, all the results of machine learning can be extended to graphs, providing a powerful set of strategies to handle this new complexity.
In this context, the problem is formulated as the analysis of communities on on-line social networks: users network structure is exploited in order to describe patterns and behaviours that would not be possible to catch using standard features. A method is developed to handle the overall analysis: specifically, the focus is to encode the heterogeneous networks, that naturally arise from the data, into simplified versions, that are able to capture all the needed information in order to effectively model the user. Machine learning tasks, such as classification and clustering, are then applied on these networks in order to show the power of the model, along with the possible new knowledge that can be extracted.

## 1.2 Structure of Thesis

The thesis is structured as follows.
In Chapter 2 the background for the comprehension of the work is defined.
In Chapter 3 works related to this thesis are presented.
In Chapter 4 there is the description of the method.
In Chapter 5 the main experiments and results are presented.
In Chapter 6 there is the conclusion of the work, including criticality, issues encountered and possible future works.

# Chapter 2

# Background

The focus of this chapter is on the background needed to support the thesis' work. First section defines an overview of the main graph theory definitions needed, then the discussion proceeds with general studies on Social Network Analysis and Graph Mining, which cover a large variety of tasks and results in literature, that are the basis the development of the thesis. Then, more specific and recent works are presented: link prediction, community detection and representation learning, which are the core elements of this work, too.
In the second part of the chapter, an extremely important result for this work, the node2vec algorithm [11], is described in detail: it is a flexible representation learning algorithm used to generate the continuous feature vectors, that are then exploited to solve a set of machine learning tasks.
Finally, the attention is put on the main technology that is transversal to the thesis, the SNAP library: it allows to handle big graphs very efficiently and it is used to build and manipulate all the networks discussed in the next chapters.

## 2.1 Graph Theory

A *graph* is a mathematical model defined as an ordered pair $G = (V, E)$, where V is a set of vertices and E is a set of edges such that $E = \{(u, v)|u, v \in V\}$. Vertices correspond to *entities* and edges correspond to *relationships* between them. Usually, a graph is represented by an adjacency matrix $A$, of dimensions $|V| \times |V|$, such that each entry is either 1 or 0, indicating the presence of an edge for each pair of nodes. A graph can be of different types and the most important categories are described:

- **undirected**: given a pair of nodes $u$ and $v$, the edge $(u, v)$ is identical to the one $(v, u)$.
- **directed**: edges have orientations and so $E$ is a set of *ordered* pair of nodes.
- **weighted**: edges have assigned a number $w_{ij}$, the weight, which can represent different variables, such as a cost, a length or a capacity, depending on the modelled domain. In this case, the adjacency matrix is extended to consider the weights and each entry becomes $a_{ij} = w_{ij}$.

- **multigraph**: *multiple* edges between each pair of nodes are allowed and *labelling* for each node and edge is introduced.
  Formally, it can be defined as an 8-tuple

  $$G = (\Sigma_V, \Sigma_E, V, E, s, t, l_V, l_E)$$

  where:

  - V is a set of vertices and E is a set of edges,
  - $\Sigma_V$ and $\Sigma_E$ are finite alphabets of the available node and edge labels,
  - $s : E \to V$ and $t : E \to V$ are two maps that indicate the *source* and *target* vertex of an edge,
  - $l_V : V \to \Sigma_V$ and $l_E : E \to \Sigma_E$ are two maps describing the labelling of the vertices and edges.

A graph has specific characteristics and some of them need to be explained, in order to better understand the work. First of all, a *path* is a finite or infinite sequence of edges that connect a sequence of nodes.
The *degree* of a node is the number of edges that are connected with the node and it can be further detailed for directed graphs in *in-degree*, the number of ingoing edges, and *out-degree*, the number of outgoing edges.
The adjacent node of a node $v$ is a node that is connected to $v$ by an edge. This definition allows to introduce one of the most important concepts in graphs: the **neighbourhood** of a node. Given a graph $G$, the neighbourhood of a node $v$, $N_G(v)$, is the induced subgraph of G consisting of all nodes adjacent to $v$.
A graph can also be associated a special *class*, if there are specific conditions. For the purpose of this thesis, two main configurations are presented. A graph is **complete** if each pair of nodes is joined by an edge, so that it contains all the possible edges. An undirected graph is **connected** if, given any pair of nodes $u$ and $v$, there is a path from $u$ to $v$, while a directed graph is **weakly** connected if replacing all of its directed edges with undirected edges produces a connected undirected graph. Otherwise, if it contains a directed path from $u$ to $v$ and a directed path from $v$ to $u$ for every pair of vertices $u$, $v$, the graph is called **strongly** connected. If only a subgraph has one of these properties, it is referred to as a *connected component*.
Finally, *centrality measures* are another core part for comprehending the main graph analysis works. They are indexes, defined and analysed widely in literature, that explains the position of each node with respect to the others, highlighting the ones that are more central. It follows a brief list of the most important ones:

- **Degree Centrality**: Given a node $v$, normalized degree centrality is given by

  $$D(v) = \frac{deg(v)}{N-1}$$

where $deg(v)$ is the degree of the node and $N$ is the number of nodes present in the graph.

- **Closeness Centrality**: Given a node $v$, normalized closeness centrality is given by

$$C(v) = \frac{N-1}{\sum_u d(u,v)}$$

  where $d(u,v)$ is the distance between node $u$ and $v$.

- **Betweenness Centrality**: Given a node $v$, betweenness centrality is given by

$$B(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}}$$

  where $\sigma_{st}(v)$ are the shortest path between $s$ and $t$ that pass through $v$ and $\sigma_{st}$ are all the shortest path between $s$ and $t$.

For a more detailed description of these and all the other metrics, refer to [9] or any other graph theory book.

## 2.2 Social Network Analysis

Social Network Analysis is a quite old discipline, which dates back to the XX century, when the term "social network" was coined. Since the fifties, the term has been used in a systematic way to indicate a kind of analysis that mixed sociology, mathematical models and, a little bit later, computer science [8]. The work of Mislove et al. [23] is one of the first that deal with modern social networks, analysing basic metrics and features for YouTube, Flickr, LiveJournal and Orkut. It describes fundamental properties of these real-world networks, making a comparison among this websites and their correspondent network structure. In the study, in-degree, out-degree, average path length and clustering coefficient are computed and compared with respect to theoretical graph models, in order to show the most important properties. Furthermore, this work makes some observations about issues that can arise in the design of the correspondent systems.
In more recent works, graphs are used also as support structure for specific tasks, such as anomaly detection in the survey of Akoglu et al [1]. They point out that anomaly detection can be better addressed if the data is structured as a graph, due to the relational nature of the problem: data are inter-dependent, exhibit long-range correlations and the fraud is often opportunistic or organized, elements that can be all captured by the network model.

## 2.3 Link Prediction

An extremely important field in graph mining is **link prediction**. The general idea is to define a similarity measure between each pair of nodes, in order to predict the presence of a link in correlation to the magnitude of the similarity. This specific problem has been largely studied and a lot of techniques are already available, as pointed out in [2, 19, 21]. The surveys describe in detail a set of possible similarities that can be selected. In this section, the most important ones are summarized.
Given a pair of nodes $(u, v)$, for *node neighbourhood based* category, there are several possibilities:

- **Common Neighbours**:

  $$CN(u, v) = |N(u) \cap N(v)|$$

  represents the size of common neighbours.

- **Jaccard Coefficient**:

  $$J(u, v) = \frac{|N(u) \cap N(v)|}{|N(u) \cup N(v)|}$$

  represents the common neighbours normalized by the total number of neighbours of the two nodes.

- **Adamic/Adar**: given a set of features $z$, shared between $u$ and $v$,

  $$A(u, v) = \sum_{z \in N(u) \cap N(v)} \frac{1}{log(|N(z)|)}$$

  which weighs more common neighbours with smaller degree.

- **Preferential Attachment**:

  $$PA(u, v) = N(u) \cdot N(v)$$

  represents an aggregated function that follows the *rich get richer* model.

For *path based* category, the main similarities are:

- **Katz**:

  $$katz(u, v) = \sum_{l=1}^{\infty} \beta^l \cdot |paths_{uv}^{<l>}|$$

  which is a variant of shortest path distance, but penalizing the longer paths.

- **Commuting Time**:

  $$C(u, v) = H_{u,v} + H_{v,u}$$

  where $H_{u,v}$ is the *hitting time*, the expected number of steps for a random walk to get from $u$ to $v$.

Other than these categories, similarities that can be used for each pair of nodes are based on nodes and edges attributes, which can improve performance in prediction, but they require some domain knowledge to be exploited. The prediction, then, is made based on the fact that the more two nodes are similar, the higher is the probability that a connection should be in the graph between them.

## 2.4 Community Detection

The idea behind link prediction is that similar nodes are connected or, at least, near in the graph structure. This concept is emphasized in another research domain, which is referred to as **community detection**. This is one of the widest and most important field in graph mining and its aim is to group together nodes that are near in the graph, leading to definition of *communities*.
Everything started from the work of Girvan-Newman[10]. They defined an algorithm based on the definition of the *edge betweenness* centrality measure as the number of shortest paths between pairs of vertices that run along it: the higher this index, the more important is the correspondent edge for community definition. In fact, if there are communities that are loosely connected, all the shortest paths will pass through these specific edges, maximizing the value of this index. So the algorithm iteratively removes the edge with highest betweenness, recomputes all the betweenness and then stops when no edges are left. The output of the algorithm is a dendogram, from which the communities can be extracted. Since this algorithm runs in $O(m^2 n)$, for recent networks, with millions and millions of edges, is computationally expensive. Another important basic result comes from the same authors in a following study [26]. In this paper there is the definition of the **modularity** function:

$$Q = \frac{1}{4m} \sum_{ij} (A_{ij} - \frac{k_i k_j}{2m}) \delta(c_i, c_j)$$

where:

$m$: number of edges in the networks,

$k_i$, $k_j$: degrees of nodes i and j,

$A_{ij}$: corresponding entry of the adjacency matrix,

$\delta(c_i, c_j) = 1$ if node i and j belong to the same community.

The function represents the fraction of edges that falls within a given group minus the expected fraction if edges were distributed at random. The value spans between -1 and +1: a high Q for a group of nodes represents a tightly connected set of nodes, thus a community. An entire class of algorithms is developed on this observation and it is referred to as *modularity maximization*: because the search space is exponential,

greedy solutions and heuristics have been devised [25, 6, 3].
Apart for modularity, many strategies can be implemented to extract communities, from advanced heuristics to algorithms more related to machine learning, that are discussed in the next chapter.

## 2.5 Representation Learning

Community detection and link prediction are strongly correlated with machine learning techniques and thus rely on user-defined heuristics to extract features encoding structural information about a graph (e.g., degree statistics, kernel functions, similarity measures). Recently, there have been an increase in approaches that automatically learn to encode graph structure into low-dimensional embeddings, numerical vectors that represent each node. Some researches have already been made [15], with a number of techniques defined to map either nodes, edges or entire subgraphs into numerical vectors with the common objective of easing the subsequent machine learning task. There is no straightforward way to encode the network structure inside a feature vector, so the idea behind *representation learning* is to learn a mapping that embeds nodes in a low-dimensional vector space $\mathbb{R}^d$, treating this step a machine learning task itself and using a data-driven approach. Most of these algorithms are *unsupervised* and they are divided into two main categories: *matrix factorization* and *random walk* based. The first are inspired by classic techniques for dimensionality reduction, which optimize loss function of the form

$$L \approx \|Z^T Z - S\|_2^2$$

where $S$ is a matrix containing proximity measures and $Z$ is the matrix of node embeddings. The goal of these methods is to learn embeddings for each node such that the inner product between the learned embedding vectors approximates some deterministic measure of graph proximity.
The other category tries to define embeddings such that nodes have similar vectors if they co-occur on short random walks over the graph and this results in a flexible, stochastic measure of graph proximity. The basic idea is to compute the probability $p_{G,T}(v_j|v_i)$ of visiting a node $v_j$ on a length-$T$ random walk starting at $v_i$, with usually $T \in \{2, ..., 10\}$. This leads to minimize the cross-entropy loss

$$L = \sum_{(v_i,v_j)\in D} -log(p_{G,T}(v_j|v_i))$$

In this environment, one of the most interesting works is the node2vec algorithm, developed by Grover et al.[11], which is selected for the development of this work and discussed in the next section.
Other works try to take advantage of the incredible amount of data available to train models based on Artificial Neural Networks. The work of Hamilton et al.[14] exploits both node structure and node attributes, feeding them to a Convolutional Neural Network. The idea is to propagate information across the graph to compute node

features: each node induces a computation graph, which is based on node features and on node neighbours features. These are combined using an aggregation function $\gamma$, that can be mean, max-pooling, LSTM . The set of parameters $W^{(k)}$ and $Q^{(k)}$ are fixed and are computed in a supervised way, leading to an extremely scalable algorithm.
Deep learning approach is also applied for dealing with *heterogeneous* networks: in this work [5], a loss function that integrates both structure of nodes and their types is defined and then it is used to train a Convolutional Neural Network architecture, composed of 5 convolutional layers and 2 fully connected layers. They take as an example web graphs composed by images and text nodes, observing that the best results are obtained when they combine both link and content features. A similar approach is used in [30], where a deep architecture is developed focusing on each node. They iterate through relationship types (both in and out-going), encode the neighbours nodes found and the resulting representation is normalized and passed through an activation function, such as ReLU. The results achieved on RDF networks are remarkably positive, outperforming the state of the art for this specific category of graphs (relational graphs). This last set of works extend the basic approach, creating embeddings directly from heterogeneous graphs, at the cost of a more complex algorithm applied.

## 2.6 Node2Vec

The result of Grover et al. [11] is the starting point of this work. The node2vec algorithm tries to solve a big problem in network analysis: feature representation for nodes and edges in a task-independent way. The algorithm develops 3 main phases in order to achieve this goal:

1. *Computation of transition probabilities for each edge*
   The weights are computed to generate a *biased random walk* of length $l$. Let $c_i$ be the $i^{th}$ node in the walk, then nodes are selected considering the distribution

$$P(c_i = x|c_{i-1} = v) = \begin{cases} \frac{\alpha_{pq} \cdot w_{vx}}{Z} & if\ (v,x) \in E \\ 0 & otherwise \end{cases}$$

   The $\alpha$ is called the *search bias* and it depends on the parameters $p$ and $q$:

$$\alpha_{pq}(t,x) = \begin{cases} \frac{1}{p} & if\ d_{tx} = 0 \\ 1 & if\ d_{tx} = 1 \\ \frac{1}{q} & if\ d_{tx} = 2 \end{cases}$$

   with $d_{tx}$ denoting the shortest path distance between node $t$ and $x$. Intuitively, parameters $p$ and $q$ describe how fast the walk explores and leaves the neighbourhood of the starting node. The variation of these two values allows to interpolate between Breadth-First Search and Depth-First Search, which are the two extremes strategies. In the specific, parameter $p$ is called the *return* *parameter* and it controls the likelihood of revisiting a node in the walk, while $q$ is the *in-out parameter*, which allows the search to differentiate between inward and outward nodes. The choice of the tuple $(p, q)$ defines the strategy $S$ of sampling the neighbourhood for each node in an extremely flexible way.

2. *Execution of the biased random walks*
   For each node $v$, a neighbourhood $N_S(v)$ is sampled, based on the nodes extracted in the walk execution, which depends on the strategy $S$ defined in the initial step. In order to offset the bias, the random walk is executed multiple time for each node.

3. *Run of the optimization problem to compute node embeddings*
   The embeddings are extracted maximizing the log-probability of observing a network neighbourhood $N_S(v)$ for node $v$, conditioned to its feature representation $f(v)$:
   $$\max_f \qquad \sum_{v \in V} log(P(N_s(v)|f(v))$$
   Assuming that the likelihood of observing a neighbourhood node is independent of observing any other neighborhood node given the feature representation of the source and that source node and neighbourhood node have a symmetric effect over each other in feature space, the final optimization problem becomes:
   $$\max_f \qquad \sum_{v \in V} [\, -log Z_v + \sum_{n_i \in N_S(v)} f(n_i) \cdot f(v)]$$
   with the per-node partition function $Z_v = \sum_{u \in V} exp(f(v) \cdot f(u))$ approximated using negative sampling. The equation is optimized with Stochastic Gradient Ascent over the model parameters to define the features $f$.

The resulting output is a feature vector $f(v)$ for each node $v$, that can be used as input to any kind of machine learning task and that is extensively used in this work.

## 2.7 SNAP

Graphs are complex and expensive objects to be represented and analysed and, recently, the dimensions of networks reach up to millions and millions of nodes and edges: an efficient library is needed in order to make analysis in this environment.
The Stanford Network Analysis Platform (SNAP) [20] is a general-purpose system that allows to easily manipulate and analyse large networks, implemented in C++, but with a Python interface (SNAP.py [34]), the one used in this work.
SNAP library is selected for this work for a number of reasons. First of all, the **performances**: the library allows to efficiently manage big graphs in main memory. For instance, a 4 GB machine is needed in order to generate and manipulate a graph of 1 million nodes and edges. This optimized representation eases the definition of local tests, without too much computational power. Moreover, provided to SNAP the edge

and node tables, little time is needed to build such networks, which makes even easier to repeat the analysis. **Flexibility** is another important point: the library allows not only insertion and deletion of nodes and edges, but also extraction of subgraphs from the original one in a very straightforward manner. This last feature, specifically, has been extensively used in the thesis work, in order to build graphs filtering the type of edges, or using only nodes that belong to a specific cluster identified.
SNAP implements 2 main types of models: *graphs* are the basic model, they can be directed or undirected and they use numeric ids to identify each entity, while *networks* allow to add attributes on edges and nodes. Moreover, there is a third model, the *multimodal networks*, that includes a modular definition of heterogeneous graphs. Except for this last one, graph models can take advantage of all the algorithms that are already implemented, which are 140: they span from basic centrality measures, to more sophisticated algorithms, from community detection to search algorithms, from neighbourhood analysis to graph model generators.
During the development of the thesis, given the possibility to use the already defined algorithm, the second model is implemented and all the possible edges and nodes types are manually defined.

# Chapter 3

# Related Work

The objective of this chapter is to describe the scenario in which this work is inserted. The main studies and papers that show similar methodologies and similar objectives are presented in detail. Specifically, the focus is about studying social media communities using advanced features extracted from the network structure, along with other features that characterize the users, in order to describe communities in different ways.
The chapter is divided in three sections, that analyse groups of different works: the ones that encode user in a specific vector to map its behaviour, the ones that exploit more the network structure and, finally, the ones that use representation learning.

## 3.1 User Features and Behaviours

Users are the most important target of all the works at social media level and there are many strategies possible to manage this core entity. One of the most used in literature is to describe the user with a set of characteristics, called *features*, over which a comparison is made. Usually, a similarity measure is then implemented in order to compare the users and group them together in clusters or classes.
The general task of "user classification on on-line social networks" very often becomes "malicious users detection", as happens in [40]. The approach is to extract a set of very specific features in order to represent the user as a vector, which is the input of the classifier that discriminate between normal users and malicious ones. Also [36] uses a combination of features in order to solve the problem of spam detection. As previously, the main features consider domain specific information for solving the task, such as presence of links in consecutive tweets. Anyway, these works define a similar machine learning task with respect to this thesis, because detecting if a user is a regular one or a spambot is comparable to discriminate between consumer and non-consumer users.
The work of Pennacchiotti et al. [29] defines a general machine learning framework for social media user classification, testing different set of features extracted from Twitter. They analyse in detail 4 set of features (user profile, user tweeting behavior, linguistic content of user messages and user social network features), pointing out

that user features alone are not enough descriptive, while content-based feature are more valuable. The work of Yu et al. [38] wants to achieve a different goal, User Identity Linkage (UIL), on social media platforms, exploiting a feature vector representative of the user. This vector is computed based on topics: each topic is a vector projected in a latent space and the users are represented in the same space, based on the relationship in time with these topics. The work of Yamak et al. [37] develops a complex framework for SockPuppets detection, exploiting both machine learning and network structure. It is different in the execution of the overall pipeline: first, specific domain features are extracted from the social media accounts in order to detect the malicious users via machine learning techniques and then two different weighted graphs are processed. The action and the relationship graphs are then used as input for a community detection algorithm, in order to cluster the users previously identified. This other set of works defines a similar approach, since the idea of encoding the user characteristics in a feature vector is exploited extensively in the thesis.
Generally speaking, there are a couple of differences between the works presented in this section and the thesis. First of all, the features extracted are extremely domain-specific, which means that they cannot be generalized, while the point of extracting features directly from the network structure is to have general features, that are able to map hidden behaviours on social media and thus be applicable for multiple machine learning tasks. Also, some of these works claim to use ”network” features, but in fact they are referring to friendship relationship, such as number of friends and followers. So, they consider only quantitative values representative of the network, not the entire social graph that is built around the users, which, instead, is extensively exploited in this work.

## 3.2 Network Features and Structure

The focus of this section is to highlight works that try to make better use of the network structure itself for describing the users.
Moving in this direction, the clustering task can be easily exchanged with the community detection task, where the position of nodes in the network gives the definition of the groups, as already described in the previous chapter. Here the aim is to present studies that use this approach in different settings and with different types of networks. [32] defines a method to detect overlapping communities in a robust way, combining tensors and ego-net in order to obtain groups that are not ”susceptible to resolution limits”. In this setting, Sheikholeslami and Giannakis extract hidden communities from networks, considering only the pure network structure. The method presented, on the other hand, is not biased toward a limit in the number of communities discovered, while is even more flexible since it considers different types of networks. [17] is another study that deals with overlapping community detection, in a complementary way, with a bottom-up technique. It devises a complex heterogeneous network, composed by both directed and undirected edges, to represent semantic interests and social interactions. They claim that the vector space representation has

weaknesses that the ontology-based one can overcome and the resulting graph is used as input for community detection algorithm: this is ”helped” by multiple seeds, that guide the definition of groups. In the work of Mosavi et al.[24], the observation is that a community is composed by a number of leaders and followers and that the first are often easier to detect, due to their strong activity inside the network. They run a *frequent pattern mining* algorithm on the action table of each user, extracting small groups of users, who are similar in term of operations they perform. Then, these groups are filtered by the ones that are not strictly correlated on the graph (some users perform the same actions but they are far away in the graph) and the communities are built, starting from these small sets of ”leaders”. A different approach is proposed by He et al.[16], who define a new method in order to discover hidden communities with respect to the dominant ones. The idea is that hidden communities are overlapped with the dominant ones, but can be highlighted when the first are ”weakened”, using different methods: *removing edges*, which removes all intra-community edges, *reducing edges*, which removes randomly some edges of the block to match the edge probability of the block, *reducing weights*, which scales the weights of edges in a community to match the edge probability of the block. In this other work [35], they try to optimize the detection using a weighted graph. The weights are computed for each edge, considering the *node attribute similarity*: for discrete attributes, the weight is the sum of shared attributes between the two nodes, while for continuous attributes, the arithmetic difference of normalized values is computed, in order to obtain a similarity measure. This approach allows to encode all the needed information for the definition of communities inside the weights, so that a simple algorithm, that maximizes the modularity, can be applied with meaningful results: the communities are sharply defined by the combination of network structure and node features. Moreover, the method is extremely simple and thus scalable to networks with millions of nodes.
These works extend the ”community detection” task, trying to consider hidden dynamics in order to define user groups with different levels of granularities. Similarly, in the thesis, the clustering approach has the same target, having a tunable number of groups that can be extracted: the further advantage, in the method presented, is that it can be applied to different types of networks. Moreover, the method presented extends the basic idea of community to cluster thanks to representation learning, which leads to the definition of sub-communities simply increasing the number of clusters extracted.
The other important element is about heterogeneous networks: usually, social network analysis is run on *homogeneous* graphs, with nodes that correspond to users and edges that correspond on specific relationships, dependent of the domain. In this work, the attention is moved on graphs that have different types of nodes, such as users, posts, tags. Shi et al [33] propose a survey on heterogeneous networks analysis, describing around 100 papers that deal with this new structure. They argue that heterogeneous networks are spreading more and more, such as social networks, knowledge graphs, gene networks, while most of the works still try to use homogeneous networks. Recently, there have been many studies that try to directly encode the rich properties of these networks inside the classical algorithms, creating new ver-

sions of classification, clustering, link prediction, recommendation, and many more. This shift is given by the fact that heterogeneous networks are becoming more and more popular and a set of applications that deal directly with these are needed. Despite this, there is no reference to the field of representation learning, which is even a newer field: the thesis tries to address the problem, combining both heterogeneous network analysis and node embeddings.

## 3.3 Representation Learning

The most recent research field, in the domain of network analysis, is referred to as *representation learning*, as already pointed out in the previous chapter.
This approach is taken in [22] and applied in Content Curation Social Networks (CCSNs), Pinterest and Huaban. In this work, the feature vector is extracted from two kind of networks, the one of explicit social relations (”follow” relationship) and the one of content-based social relations (”re-pins”) and then they are combined in a unique representation, used for recommendation tasks. This is one of the most similar works, the main difference stands in the construction of the networks: in this work, there is no intermediate step, because all the interactions are already encoded as direct user relationship. Instead, in the thesis, a further step is required to simplify the heterogeneous networks, before generating the embeddings. Moreover, these are built using a single optimization problem, while in the thesis the optimization is kept separated for each network.
[12] has a different target, sentiment analysis, but the approach to achieve it is quite similar to the thesis one. In fact, the set of users, products, words used in reviews and polarity of the words are included in a single heterogeneous network: in this way, using representation learning, these different entities are mapped into the same latent space. The main advantage is that the context of a word contains also the users that used it and the products that it is associated with, thus having richer description, which can be used to accurately describe the interactions between entities.

# Chapter 4

# Main Idea

In this chapter, the main method for analysing the networks is presented in detail. The core idea is to exploit the network structure in order to map the behaviour of users extracted from on-line social network communities. This structure is intrinsic in social media domain, but its effective definition and construction is not trivial and it is part of the knowledge discovery process: the main problem is dealing with the variety of networks that can be extracted. In this work, the analysis are focused first around the concept of *brands*, which use their accounts with specific purposes and behaviours on social networks, then the perspective is moved on their *communities*, where the users that follow the brand are grouped and described. A methodology is developed in order to handle these bigger heterogeneous networks and to transform them in a reduced version, so that the definition of the embeddings is easier and more significant. After this, different set of features are tested, in order to solve both a classification and a clustering tasks, verifying the different sets of information that can be extracted.

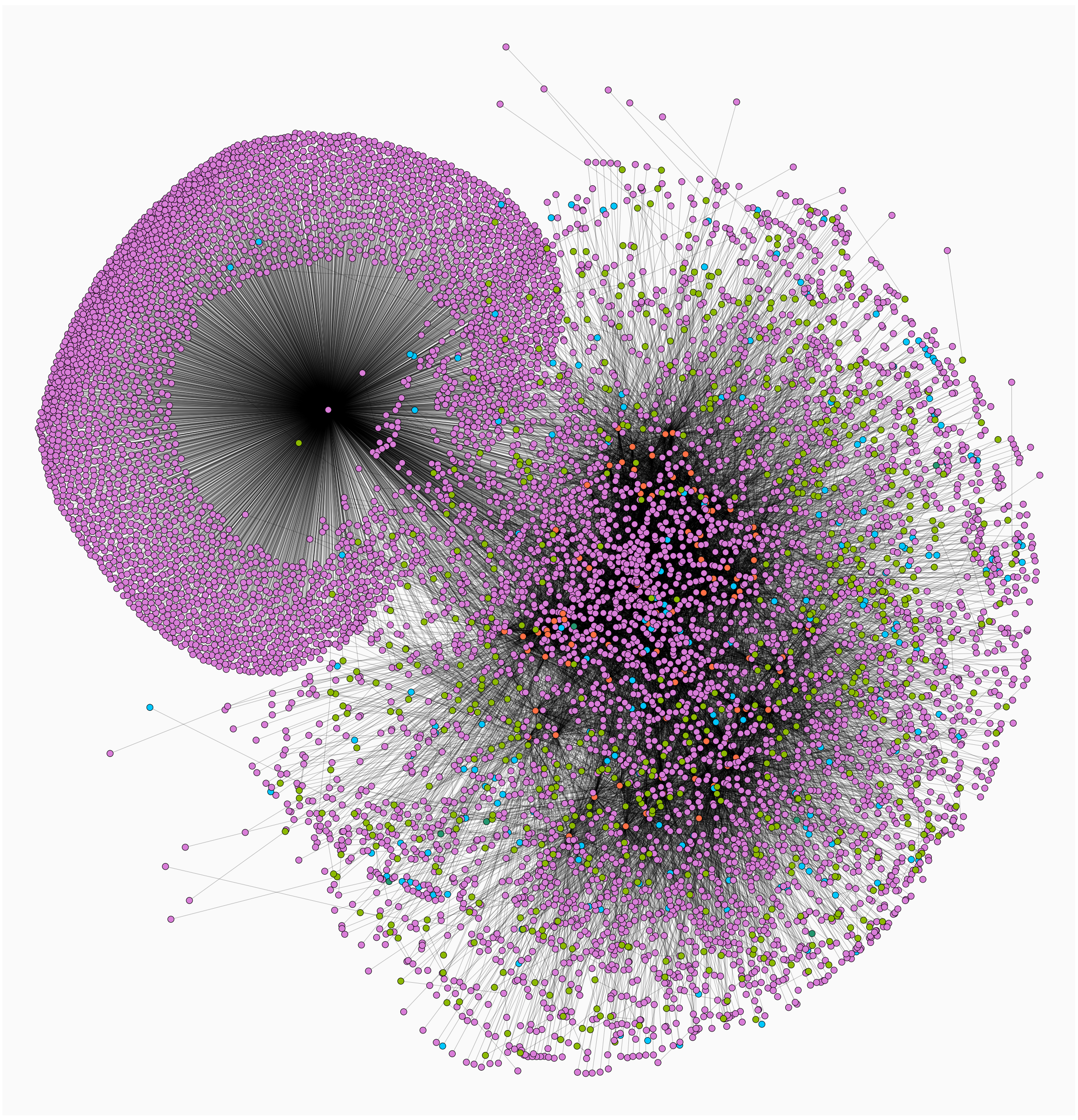
## 4.1 Brand Networks

Brands can take advantage of on-line social network in different ways, from generating content and check the almost immediate feedback that other users give, to verify its social influence with respect to the competitors. In this setting, the aim is to build networks that are able to catch such information, which can be useful to the brand's strategy.
The brand network can be a very interesting point of view: its construction is based on all the content generated by the brand itself, including the connections with other users. In particular, the main components of the network generated are:

- Brand user node, connected with all its posts
- Location and comment nodes, connected to the correspondent posts
- Tag nodes, linked to all the posts and comments in which they are used
- Other users nodes, which can be users mentioned by the brand or tagged directly on the picture, users that commented or liked a post and users followers of the brand

The resulting network is directed and heterogeneous, including different kind of nodes and relationships. An example of this model is given in figure 4.1: different colours denotes different node types. In figure 4.2, the edge types are highlighted for the same network.

Figure 4.1: Brand Network - Node Types



The figure represents the network of a single brand, where the violet nodes are the users, the orange ones are the posts, the light blue ones the hashtags, the green ones the comments and the dark green the locations. It is possible to see the brand, as the central node around which a lot of other user nodes are connected, the *followers*.

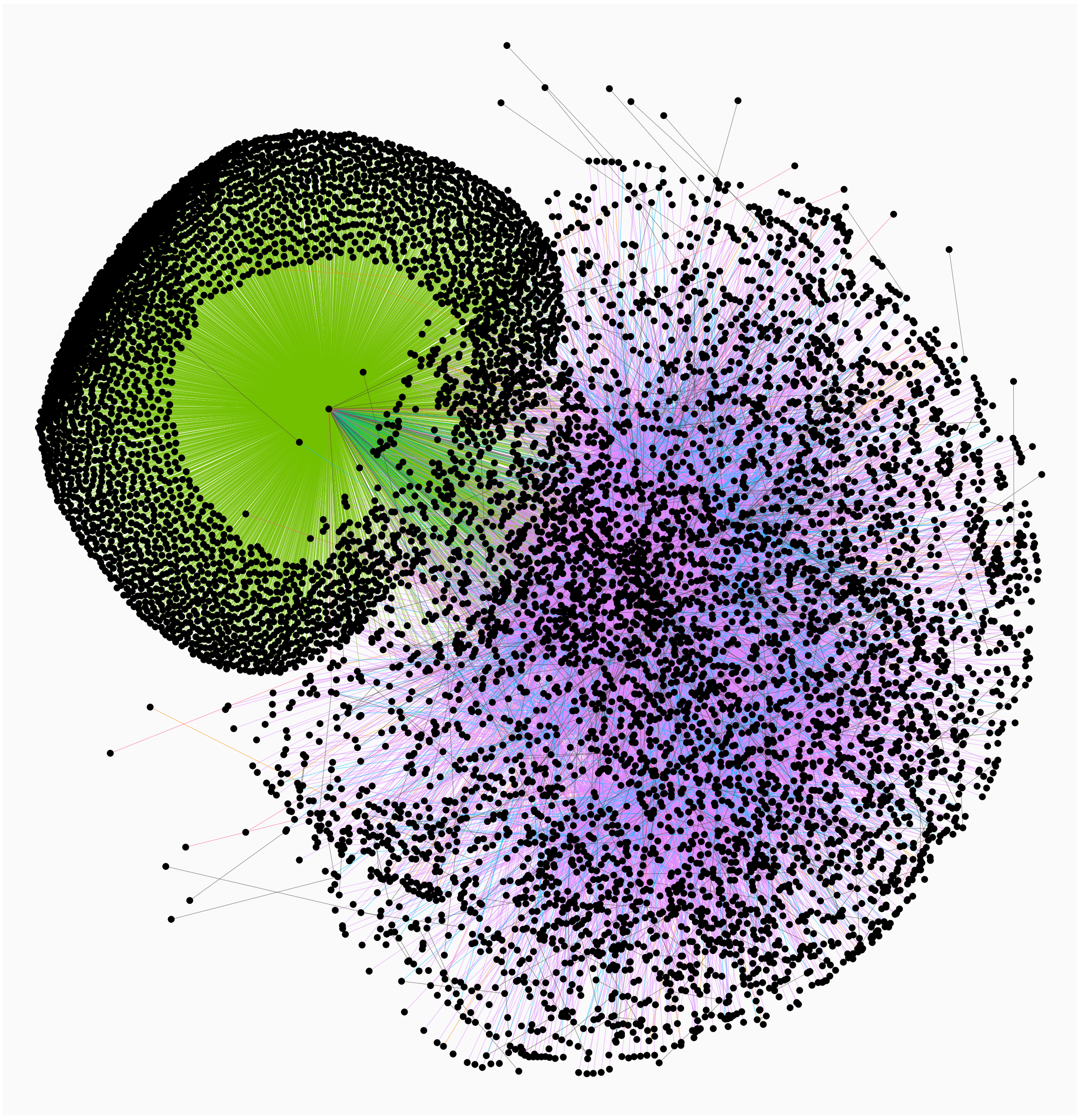


Figure 4.2: Brand Network - Edge Types

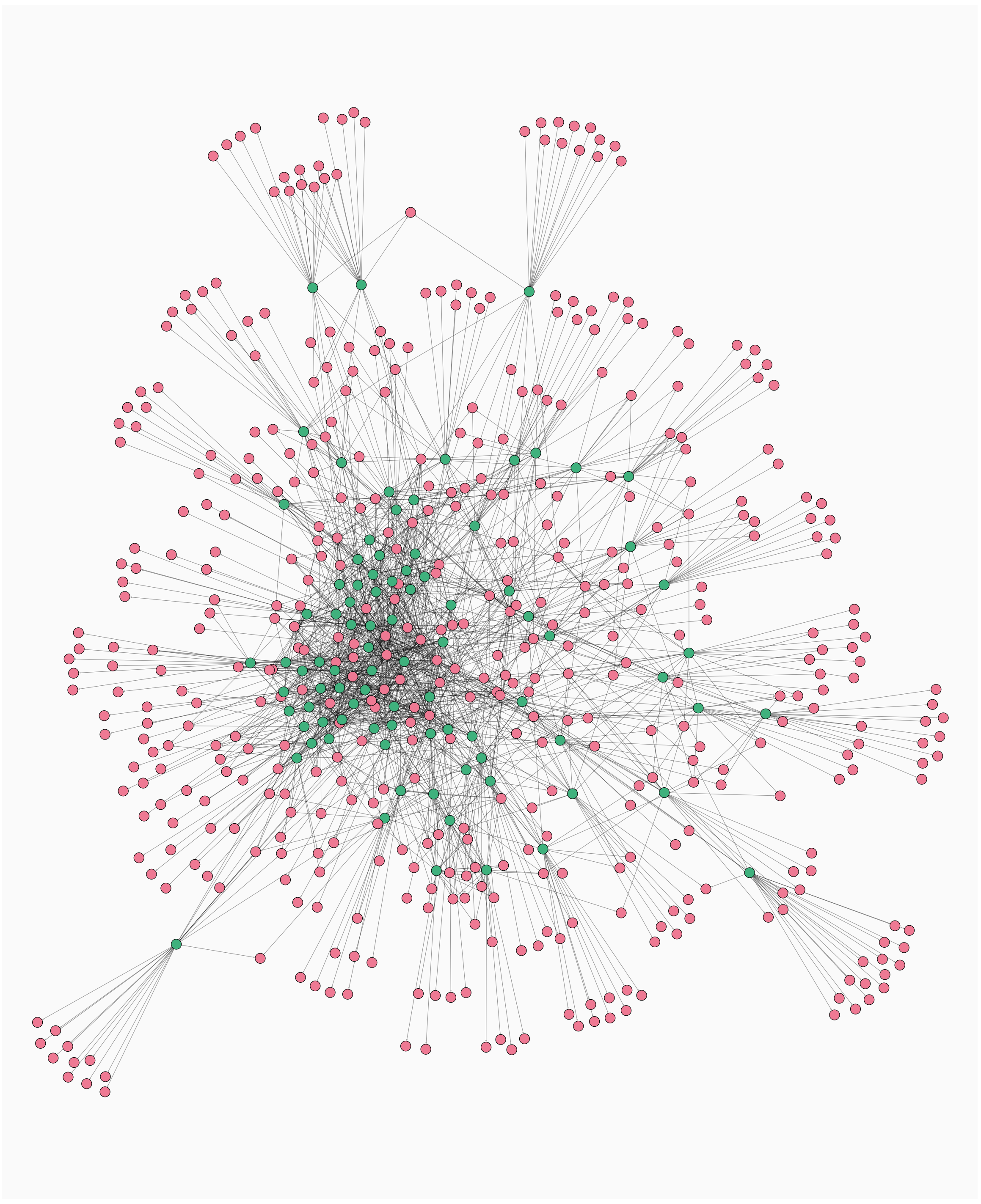
The figure shows the different types of relationships. The most visible are the most frequent, that include the like (violet), the follow relationship (light green), and the comment (light blue). The first two relationships alone cover around 90% of all the edges of the graph. The other types are author, commenter, tag, mention, picture tag and location.

This representation allows to observe the network topology of each brand, highlighting the dimensionality and importance of nodes inside its own model. Moreover, it is possible to *merge* this networks and to build a **domain** network, which models the brand content and the competitors content together. In this way, the importance of each node, either user or hashtag, is computed inside the complete domain, giving insights about who are the most popular brands, what are the most relevant topics and the niche ones.

### 4.1.1 Image Concepts Networks

An extension to the previous model is given by the inclusion of derived information as nodes of the network. In the context of social media, *images* play an extreme important role: regular users usually share images that are very interesting, because they contain places or things that the user enjoys. On the other hand, brands have the same potential: use powerful and catching images to impress possible customers about their products. For these reasons, image tagging nodes are added to the network, linking, by consequence, posts that share similar pictures. In order to obtain these taggings, Clarifai API [39] are queried, using the ”general” model: the tags retrieved focus on the main concepts of the picture, such as the presence or absence of people, the definition of a specific landscape and many more.
As previously, the network shows the most important topics, with respect to the images that are used by each brand: the importance can be either defined as the most used pictures, as the most shared or as the most niche, which can be easily extracted with centrality measures. This gives more specific information on the type of messages that a brand and its competitor want to share. Moreover, both the single brand network and the domain network can be defined, while for simplicity only the first is presented (figure 4.3). It is evident that there are concepts that fall apart in the graph, the ones very particular, and then there others more densely connected, since represent the most used images for the brand. For instance, in the middle of the graph there are concepts such as ”people”, ”woman”, ”travel”, while in the lower corners there are two very specific images: the left one has ”bath”,”shower”,”wash closet”, while for the right the concepts extracted are ”weapon”,”rifle”,”protection”. The main problem about these tags is the fact that they are very general, while the more specific filters that the API expose are not so accurate: in some context this could not be a problem, but moving on the case study of fashion, this becomes a critical issue, that cannot be ignored and the validity of the process becomes questionable.

Figure 4.3: Image Concept Network

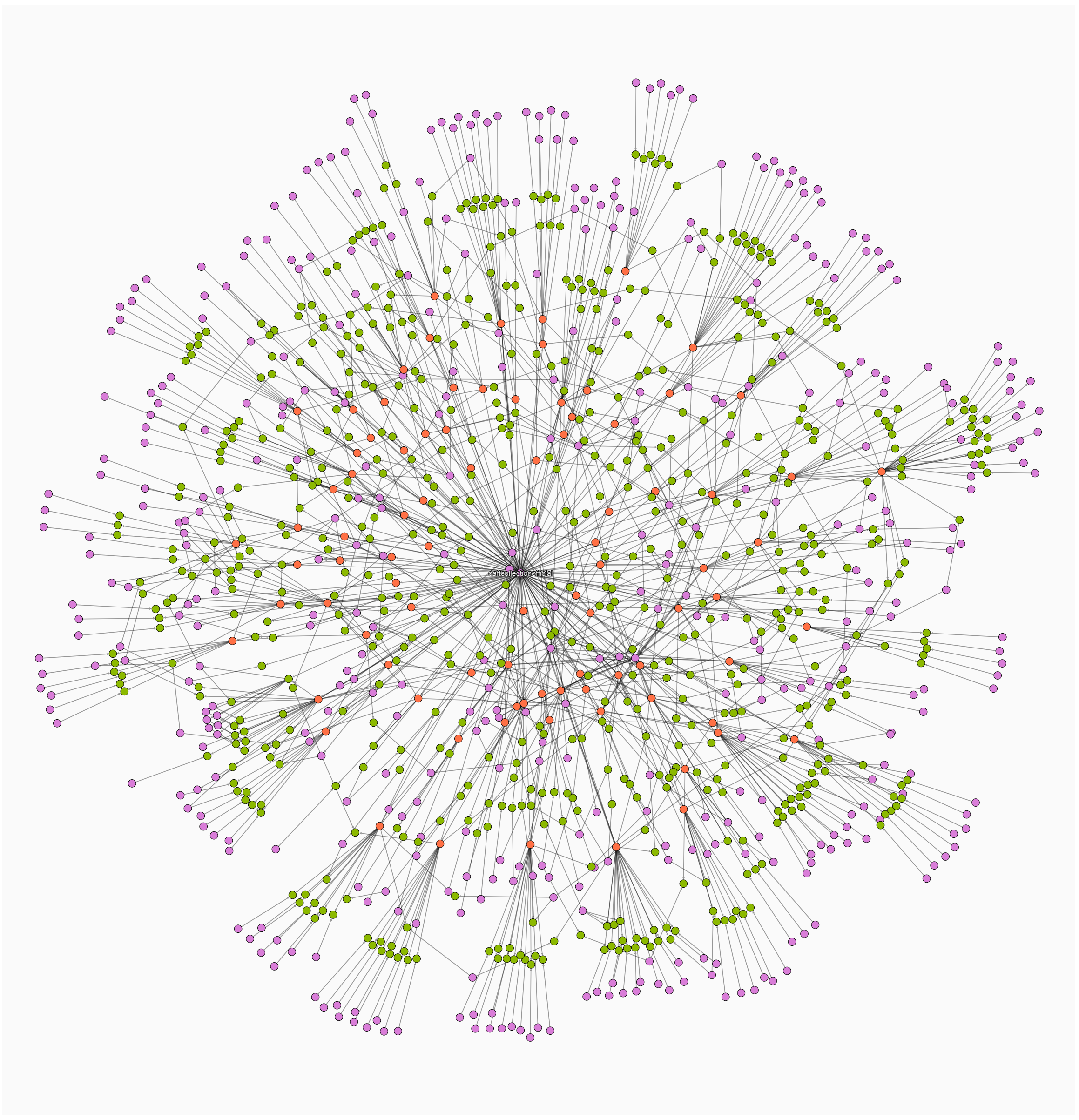


The graph in the figure represent the image concepts network of a single brand. The posts are defined in green, while in rose there are the picture tags.

### 4.1.2 Mentions Networks

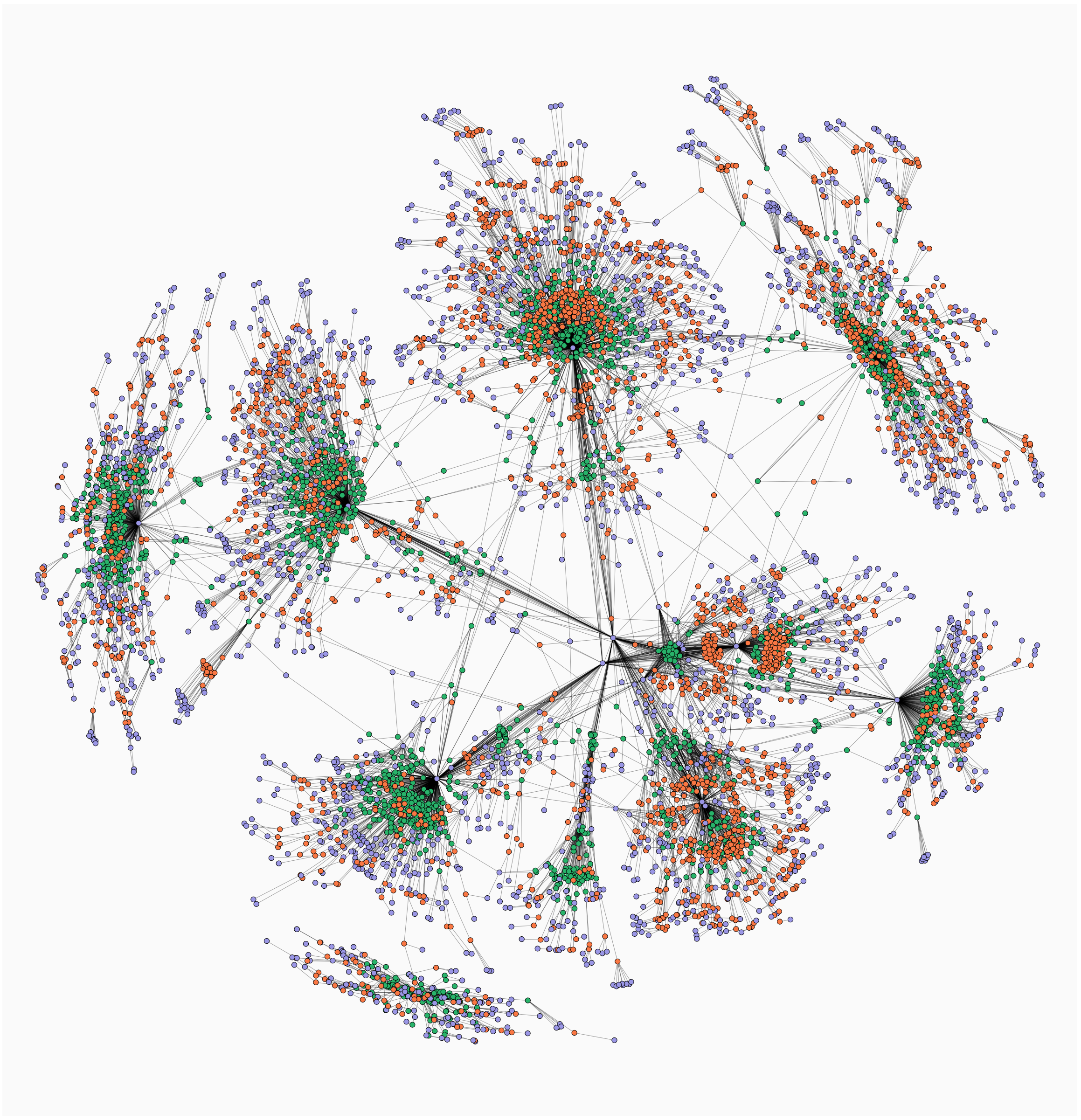
A technique extremely useful to simplify the overall complexity of this kind of models is the reduction of dimensionality. Networks are non-euclidean models, so the process is not trivial: a possible approach is working on the type of relationship, filtering only the ones that are useful for more detailed analysis.
Among all the possibilities, one of the most promising relationship is the ”mention”: it is implemented as a special tag in the post content, using the ’@’ character, and it allows to notify another user about the specific post. In a very similar way, it is possible to insert the target username directly on the picture, with the same purpose. In the work developed, these two modalities are treated as one, referred to simply as *mention*. It is a valuable information because it connects two users not due to its content, either in the picture or in the hashtags used, but due to a voluntary action, thus creating a strong link between them.
Mentions networks are built filtering the complete networks described in the previous section, keeping brand node, posts and comments nodes, and the correspondent mentioned users. The same network of figure 4.1 is filtered and represented in figure 4.4.
This filter helps handling the dimension of the domain network, defined in the previous section: figure 4.5 shows the combination of all the mentions networks of brands in a unique model. The immediate insights that can be made are that brands create sharply defined *communities*, detectable even visually, and also that there are ”bridge” nodes, which are users mentioned by (or that mention) multiple brands.

Figure 4.4: Brand Mentions Network



The network is filtered so to keep only the needed content: violet nodes are still the users and the brand can be noticed in the middle of the graph, while around it there are its posts (orange), comments (light green) and all the users mentioned in both these entities.

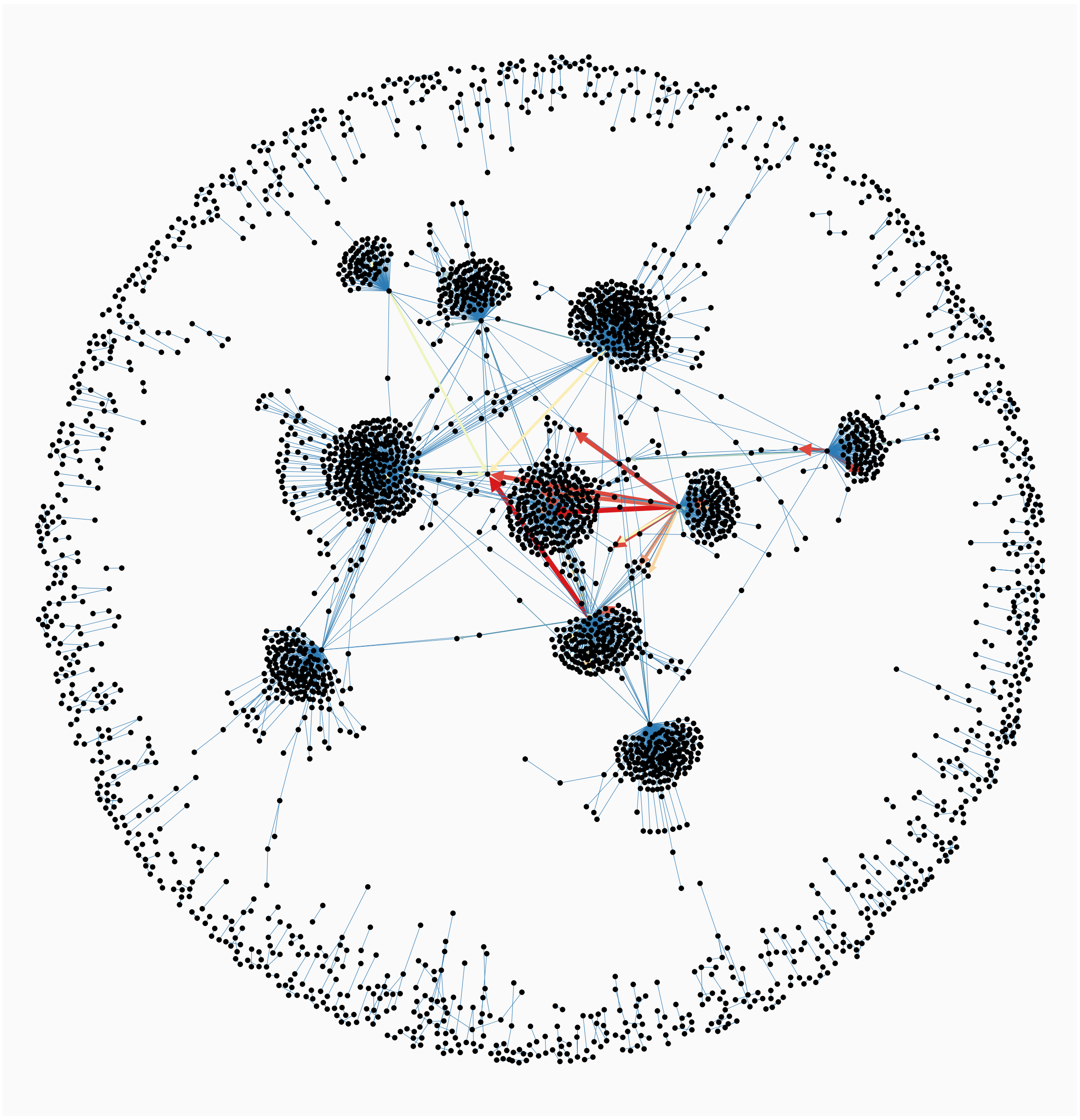
Figure 4.5: Domain Mentions Network



The graph represents the dynamics generated by the mentions of each brand: in blue there are the user nodes, in green the posts and in orange the comments. It is possible to highlight a small set of users that is mentioned by multiple brands, while most of the other users have connection with only one. This fact generates the visible densely connected components, connected to each other by few, shared nodes.

In order to move the focus on users, the last network presented is further simplified, in another reduction step: all the post and comment nodes are removed and in its place a single weighted edge $e_{ij}$ is defined, where weight $w_{ij}$ is the number of times user $i$ mentioned user $j$, independent of the fact that mention happens inside the post or inside the comment. Then, a very interesting model is retrieved, because it puts the attention on the set of users that compose the communities, generated by the mention relationship around each brand (figure 4.6).

Figure 4.6: Reduced Domain Mentions Network



The colour of the edges indicates the weight, from blue (low number of mentions) to red (high number of mentions). The sparse nodes around the main graph are users that mention other users in the comments, not correlated at all with the brand. This is a way, in social media, to show to a friend a specific content of another account.

### 4.1.3 Other Networks

The filtering possibilities for the original brand network are many: basically, each type of node and/or relationship can be kept so to obtain a different ”view” of the same network, with, possibly, different insights.
Among all the possible entities of the main network (figure 4.1), location cannot be considered, since each brand uses the ones that better match the company’s view and very little is shared, leading to an inconsistent definition of graph. The same reasoning can be applied to brand hashtags. The focus still remain on users: as for the other possibilities, likes, comments, and followers networks are extracted with the same procedure described in section 4.1.2, filtering only the nodes and edges of interest. The analysis that can be run are still the same, but the sparsity of the data does not allow to extract valuable information.

## 4.2 Community Networks

The community that grows around a brand has a significant impact: knowing who is interested in your products and what are the other interests of these users can help in better defining your strategy, creating something that is appreciated, competitive and innovative at the same time.
Community networks can be defined similarly to what described in section 4.1 for brand networks construction, with some constraints about the types of nodes included, which depends on the data available, explained in section 5.2. The dimensions of these networks are incredibly high and so two different ”versions” are presented.

### 4.2.1 Complete Networks

*Complete* networks are built considering all the posts, hashtags and users mentioned of a brand’s followers. The number of nodes and edges for the communities analysed are presented in table 4.1.

Table 4.1: Community Network Dimensions

| brand | \|V\| | \|E\| |
|---|---|---|
| emporiosirenuse | 510767 | 1892927 |
| daftcollectionofficial | 833667 | 3094603 |
| athenaprocopiou | 2135161 | 9667715 |

The three communities analysed in detail are presented with their complete graph dimensions.

It is evident that reduction is needed more than previous case and two main networks are extracted, keeping only the correspondent edge types and nodes: mentions network and hashtags network. This allows to focus the attention either on content shared by followers or on people with whom followers make contacts.

Since in this setting the users are very important from the brand's point of view, because they are followers and thus interested in brand's products, the objective is trying to extract similarities among these users, given their position inside the community network. In order to compute similarity, a representation of the user is needed and the node2vec algorithm is applied for this purpose: it is run on the two networks, in order to learn a representation of the user nodes, mapping both the interactions based on shared content and based on mentioned common users. Then, the features are normalized and combined in order to define a unique representation of the user, that encodes the main network aspects already described.
Given this feature vector, it is possible to solve machine learning tasks that allow to verify the validity of this representation. At the same time, they help in defining models useful for grouping similar users together, understanding better what is the underlying structure of the community.
A classification task is developed using these features, following a simple pipeline:

1. Definition of a ground truth set of users
2. Feature extraction and combination
3. Model selection to define the best hyper-parameters
4. Evaluation of the model

Moreover, a clustering task is developed following a similar process: extraction of features and combination, evaluation of the clusters structure, validation of clusters content. The main difference is that, given the unsupervised method, feedback by domain experts is required and to ease this step a set of labels and a ranked degree of similarity are computed.
The details of these steps and of the experiments are described in chapter 5.

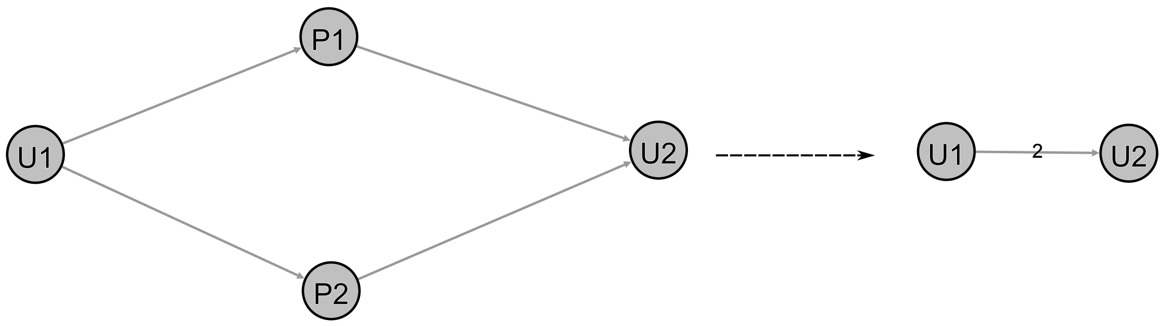
### 4.2.2 Reduced Networks

*Reduced* networks are built observing the previously presented network in figure 4.6, where only the user nodes are kept. The idea is to synthesize these enormous heterogeneous networks in smaller, weighted and homogeneous ones, which brings several advantages:

- The lower input dimensions let the manipulation of the networks to be easier and the computation to be faster.
- Keeping only user nodes allows to work effectively with "classical" social networks, where only the user-user interaction is represented, thus opening all the results already present in literature.
- A weighted network is a compact and very strong representation, which is exploited to extract hidden communities.

Two different algorithms are developed in order to obtain these networks, since hashtags and mentions have different characteristics.
For mentions networks it is simpler: the idea is to remove all post nodes and to define in its place a single weighted edge $e_{ij}$, where weight $w_{ij}$ is the number of times user $i$ mentioned user $j$, in a similar way to what described in section 4.1.2. Practically, the implementation uses the complete mentions network as input, iterates over the user nodes and, for each other user node at distance 2 in the network, which corresponds to the mentioned user, it computes how many paths are present among the two: this set of weights are stored and they become the weights of the edges between all pair of users, thus defining the new reduced network. The diagram 4.7 explains practically the transformation.

Figure 4.7: Mention Network Reduction Process



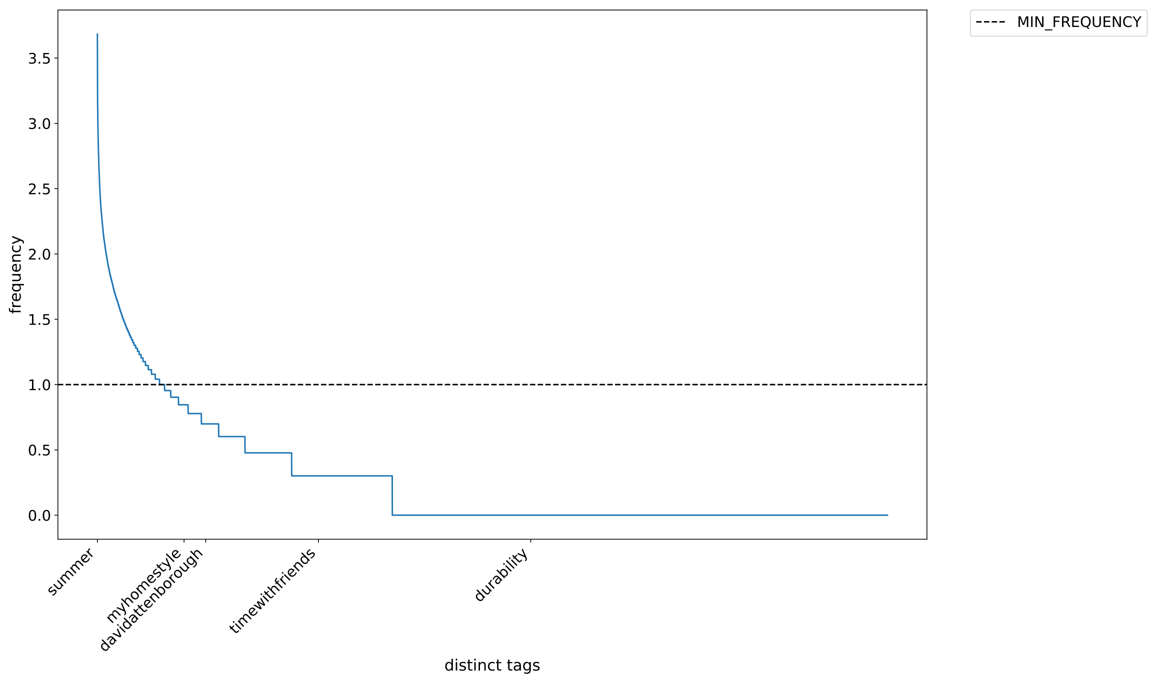
For each pair of user, the nodes between the two entities are removed and, in their place, a weighted directed edge is defined

The reduction of hashtags network is quite similar, but a pre-processing step is required: tags, despite of mentions, are massively used on social networks and connecting each pair of users that have at least one tag in common means creating a fully connected graph, where every node has an edge with every other node. Three step of filtering are applied to the complete hashtags network:

1. **Stop Words Pruning**
   There is a set of hashtags that are extremely common on the case study platform, Instagram, such as #picoftheday, #like4like, #nofilter and many more, and a set of hashtags diffused in the case study domain, fashion, such as #outfitoftheday, #ottd, #fashion. These 38 tags are completely removed from the source network, because they create a lot of connections that do not increase the effective similarity between users.

2. **Tail Pruning**
   In this step, hashtags that have a low frequency are pruned, as shown in figure 4.8. This allows to reduce the overall number of tags over which running the effective analysis, removing further noise from the network.

Figure 4.8: Tags Frequency Distribution



In figure, the frequency of each hashtag inside a community is presented: on the x-axis there are the hashtags, while on the y-axis there is the *logarithm* of the frequency for each of them. The minimum frequency considered is showed as the horizontal line, correspondent to 10 in linear scale.

3. **TF-IDF Variation**
   Term Frequency - Inverse Document Frequency is a technique very well known in data mining literature, used to map a text, described by a set of words, into a numerical vector. Since hashtags are words, it is possible to compute the weights defined by the algorithm in a very straightforward way: the idea is to consider a user as the text, while the hashtags used by that user are the terms that describe it. The computation of the IDF value for each tag $t$ becomes

$$IDF(t) = \frac{\log_2(N)}{U}$$

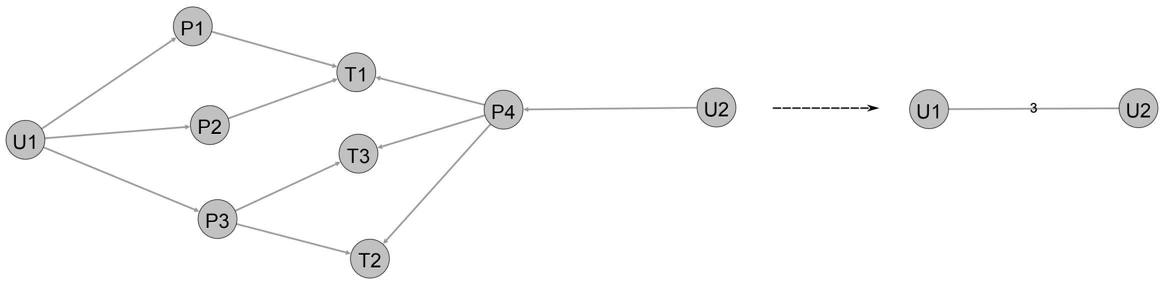
   where $N$ is the number of users inside the community and $U$ is the subset that used tag $t$ at least once. Symmetrically, the TF value for each tag $t$ used by user $u$ is the total number of hashtags used by $u$: the set of TF values for each user is extracted traversing the graph. The final weight is obtained for each tag of each user, following the basic definition:

$$TFIDF(t, u) = IDF(t) \cdot TF(t, u)$$

   This value is then used for the final filtering step: only the first **top K** hashtags, based on TF-IDF value, are kept for each user. In this way, two users are connected if and only if they have at least one of the *most important* tags in common, giving a stronger measure of similarity. $K = 17$ is selected, which is the value obtained analysing the distribution of hashtags in the communities considered in the experiments. It is showed that it is constant for all the three communities and it represents the median value of hashtags per user, normalized by the logarithm of the community dimension: $K = \frac{2 \cdot median}{log_{10}(N)}$.

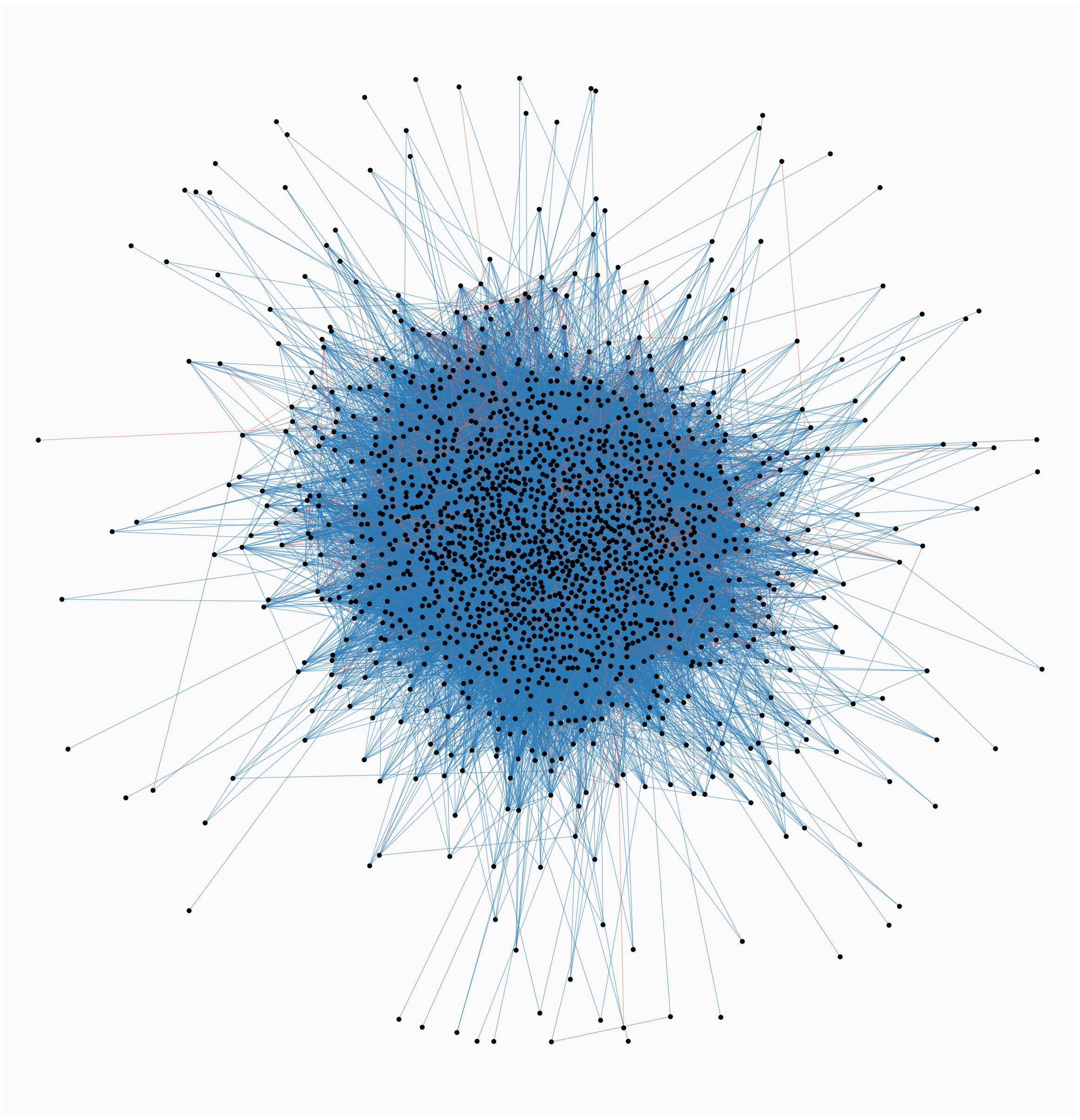
After the application of these 3 steps, the complete network is effectively reduced: for each user node, the set of hashtags used is stored, navigating through the edges of the network and then the weight for each pair of users is $w_{ij} = |T_i \cap T_j|$, where $T_{i,j}$ are the set of tags for user $i$ and $j$. As before, in figure 4.9 is represented an example of reduction, while in figure 4.10 there is a hashtags reduced network of an entire community.

Figure 4.9: Hashtags Network Reduction Process



For each pair of users, a weighted undirected edge is defined, replacing all the tag nodes between the two entities.

Figure 4.10: Hashtags Community Reduced Network



The density of connections for this kind of networks is very high, but the colors show that most of them are weak: in fact, the number of hashtags in common spans from blue (low) to red (high).

It is clear that this kind of networks is much more complex to handle, since it seems that everything is connected and no underlying structure is present, but the experiments prove the contrary.
In order to give an idea of the dimensionality reduction produced by the algorithms, since it is not feasible to produce a clear visualization of the complete community network, table 4.2 shows number of nodes and edges both for mentions network and

hashtags network, comparing the reduced and complete versions. It is clear that the reduced networks can take an incredible advantage in terms of computational time and, on the other side, little information is lost, since the weights encode most of the data needed to measure similarity.
These networks are used as input again for the node2vec algorithm and the same exact procedure can be reproduced, leading to the definition of the user feature vector and to the resolution of the same machine learning tasks, which are extensively discussed in the next chapter.

Table 4.2: Comparison between complete and reduced community networks

| | | Complete Network | | Reduced Network | | % Reduction | |
|---|---|---|---|---|---|---|---|
| brand | type | \|V\| | \|E\| | \|V\| | \|E\| | nodes | edges |
| emporiosirenuse | tag | 473128 | 1754613 | 1598 | 56551 | 99.66 | 96.77 |
| emporiosirenuse | mention | 328375 | 425495 | 44416 | 57853 | 86.47 | 86.40 |
| daftcollectionofficial | tag | 788139 | 2931279 | 2342 | 96463 | 99.70 | 96.70 |
| daftcollectionofficial | mention | 567467 | 681054 | 47447 | 59078 | 91.63 | 91.32 |
| athenaprocopiou | tag | 1993850 | 9110931 | 8847 | 1048551 | 99.55 | 88.49 |
| athenaprocopiou | mention | 1523437 | 1912101 | 148957 | 222309 | 90.22 | 88.37 |

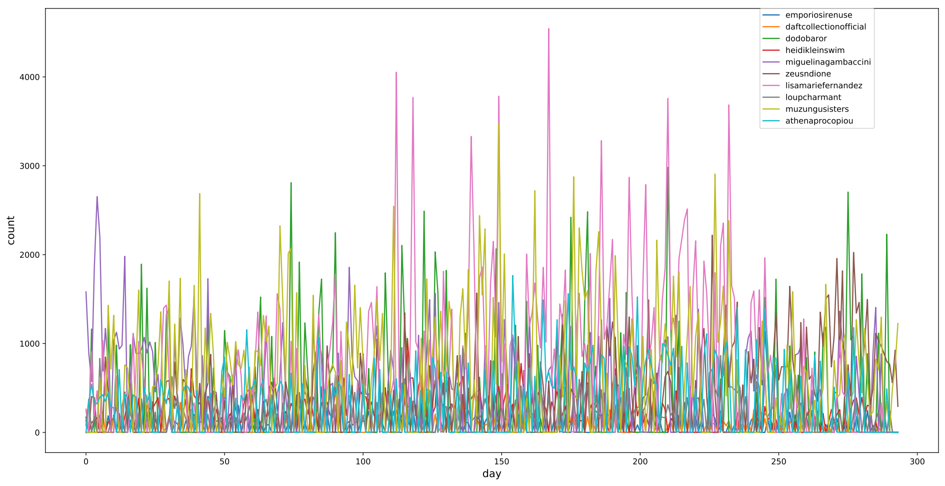
The table shows, for each community analysed, the reduction in the number of edges and nodes obtained from the defined process: it is evident that eliminating more than 90% of nodes and edges can significantly decrease issues in the overall process.

## 4.3 Dynamic Networks

Generally speaking, time is often a key component in social media, since distribution of posts and interactions can vary during different periods of the year and they can have a valuable meaning in many domains. On the other hand, this kind of data is complex to handle, because a further information, *time*, needs to be encoded in some way inside the graph structure, leading to the definition of *dynamic networks*.
Actually, there are many ways to include time inside a graph structure.
A simple solution is to consider the date and hour of posts and to build the network limiting the data to a single day or a single month. In this way, the evolution of the network itself can be tracked, highlighting specific patterns, such as the growing of communities around the brand.
Another, possibility is to use a side project of SNAP that allows to combine the time-series and the graph analysis together, the Time-Varying Graphical Lasso [13]. The algorithm takes as input a set of raw time-series, which correspond to the target entities of the study, and it estimates a set of sparse time-varying inverse covariance matrices, which describe the correlation of these time-series in specific time windows. The matrix can be used to devise an induced and dynamic graph structure, showing hidden relationships among nodes. In social network domain, time-series can be built

starting from the users' *timeline*: the sequence of posts and comments of each user can be put together to derive a temporal description of the behaviour of the user itself. An example of raw time-series, based on the posts of the brands in the case study, is represented in figure 4.11.

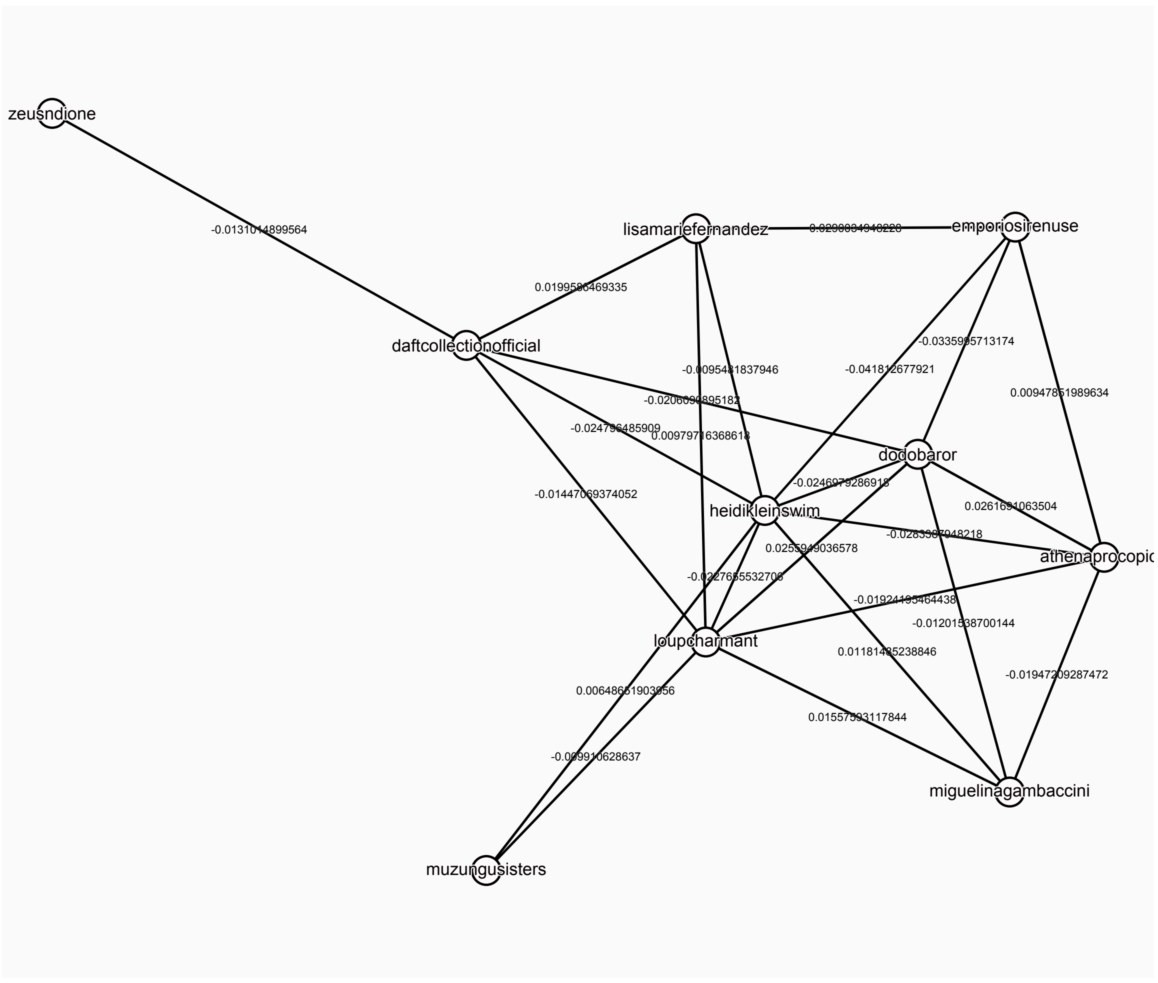
Figure 4.11: Brand Timeseries of Posts



In this figure, the value is not given by a simple count, but is weighted based on the number of likes of each post, in order to have a more variated dynamic.

It is clear that the sparsity of the posts during the months makes this kind of analysis weaker, since the time-series definition itself is weak and the added weighting cannot entirely fix the problem. Anyway, the algorithm is run against this dataset and an output graph is presented in figure 4.12.

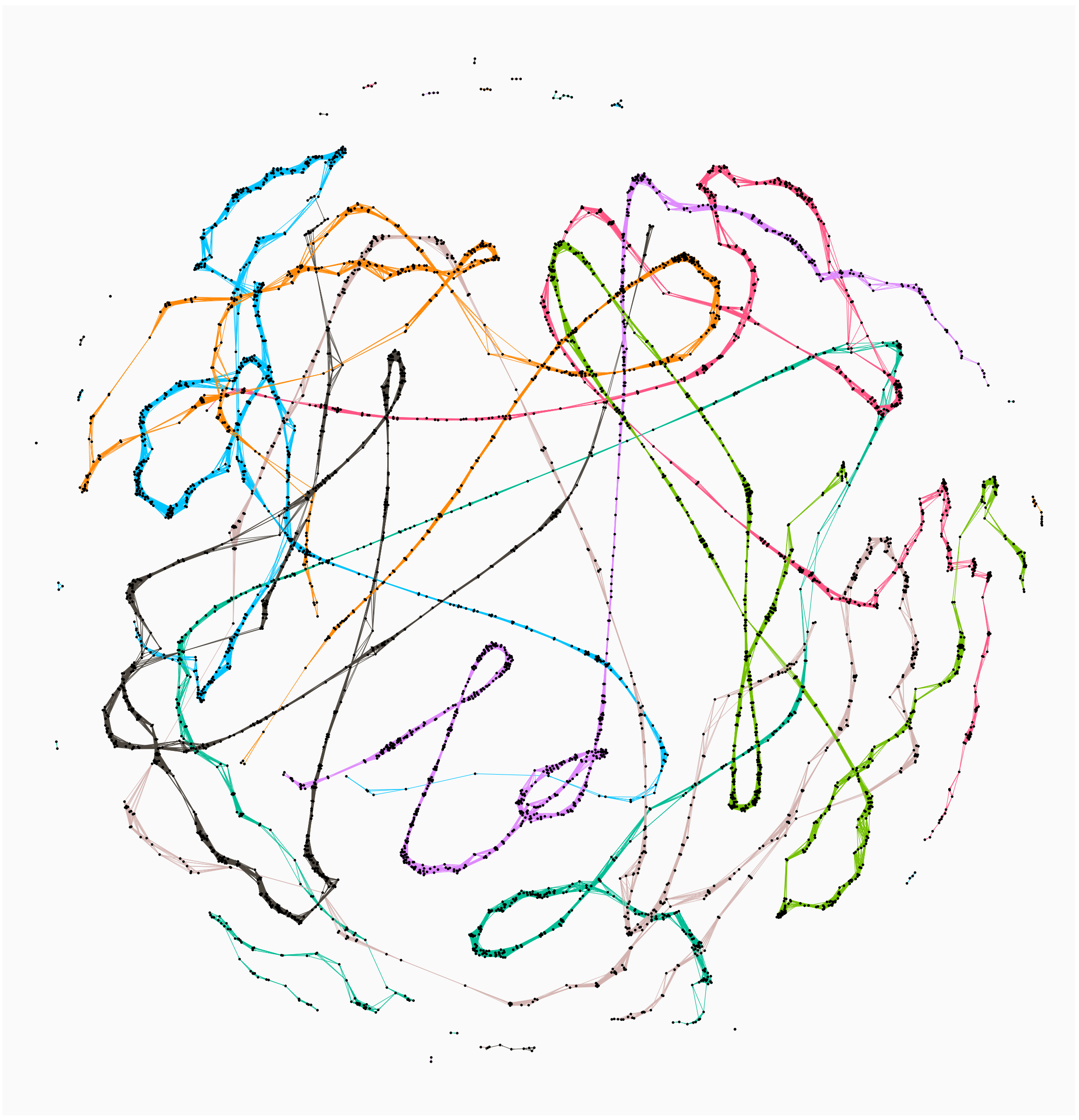
Figure 4.12: Induced Dynamic Graph Example



The weights represent the *inverse covariance* between each pair of timeseries: if this value is 0, the edge is not present.

Finally, another possible approach is to extend the definition of a network, including a *temporal edge*: this edge can connect entities that are related within a specific time window, that can be minutes or hours. Selected a suitable time window, clusters of nodes based on time could be extracted. Because of computational issues, in the case study it is not possible to apply this technique, due to the exponential number of edges that are created in the tests run. In figure 4.13, a test generated in another domain is presented to give an idea of the possible graph output.

Figure 4.13: Temporal Graph Example



The colours represent the different days of the test: each group of posts is connected to the nearest ones in time and groups of posts densely connected are visible.

The increase in connectivity around certain periods of time may indicate a specific behaviour, that could be further analysed: it could be possible to build a graph like this for each brand and then check for similarities or specific points of overlap. The biggest problem is that the generation of these edges grows exponentially and a more accurate study is needed in order to discover the correct time distances to extract significant patterns.

# Chapter 5

# Experiments

In this chapter the experiments run on the case study are showed in detail.
Firstly, there is the description of the case study itself, which is about *emerging fashion brands* and their communities. Then, the correspondent data collection process is presented: both the brands and followers public profiles are scraped, in order to gather the information needed to build the networks.
In the other sections, the main analysis are discussed along with the results. The classification task shows that the features extracted from the reduced networks are representative of the user behaviours and they are compared with quantitative features, obtained from user profiles. In the second experiment, set of followers are grouped together in clusters in order to define meaningful sub-communities, that are then validated with domain experts.

## 5.1 Case Study

”Emporio Le Sirenuse” is an Italian fashion brand that produces mainly swimwear and summer dresses, founded in 1990 by Carla Sersale. As their site points out ”Le Sirenuse, Positano, Collection is a cool Mediterranean sea breeze, a confident sashay from beach to bar, a sweet taste of India garnished with a tangy splash of Positano chic” [31]. The case study focuses on this specific brand, its community, defined as the set of followers on Instagram, and its competitors selected by domain experts: ”Miguelina Gambaccini”, ”Muzungu Sisters”, ”Daft”, ”Athena Procopiou”, ”Zeus+Dione”, ”Loup Charmant”, ”Dodo Bar Or”, ”Heidi Klein” and ”Lisa Marie Fernandez”. They are all defined as *emerging* brands, so they are trying to make their way through the fashion market, focusing on defining and sharing its own identity.
Fashion domain gets a lot of interesting insights from the social media, in the specific from Instagram: images are the core element that both domains share and that can be exploited in many ways. The main research questions are referred to extract information about *lifestyles* that are more interesting for the users, so that the brand can match them with its products and reach a broader number of people: ”what other things do people that follow my brand like?”, ”what do people that follow competitor brands like?” are schematic possible question that any brand can try to answer in

order to improve its marketing.
In this setting, the thesis work exploits extensively the network structure of these brands, highlighting similarities, differences and recurrent patterns in many different aspects. Successively, the focus is moved towards the communities, defining models that allows to characterize in a detailed manner the different groups of people that are interested in the brand, always keeping the network approach in all the steps.

## 5.2 Data Collection

The thesis case study dataset is based on a database of posts and users obtained directly from Instagram, using a manually implemented *scraper*, which is run in parallel with two Azure Virtual Machines provided by the Politecnico di Milano.
The process is divided into 4 main steps:

1. Collection of the data about brands official Instagram accounts and all its posts.
2. Collection of additional information for the posts gathered in step 1: tags on pictures, geolocation, list of comments, list of likes.
3. Collection of the followers' list for each brand.
4. Collection of accounts data and posts for each user obtained in step 3.

Moreover, a final step is later added, moving all the data inside a MySQL relational database, that allows to remove consistency errors and that can be used as the basis for the analysis. In fact, the data collected is extremely rich and all the entities considered have a number of attributes stored along them:

- **user**: user id, username, number of followers, number of following, number of posts, profile picture, privacy flag
- **post**: post id, timestamp, content, owner id, user mentioned, hashtags, number of likes, number of comments
- **comment**: comment id, content, owner id, timestamp, hashtags, user mentioned

A temporal constraint is needed to work within the same time window for all the users: posts retrieved belong to the period between $1^{st}$ January 2017 and $1^{st}$ November 2017. Another element to point out is that the data available is only for the users who have a public profile, indicated by the "private" flag set to *false*, otherwise only little data is stored. Also, for time issues, the additional information, such as comments, likes and geolocation, are stored only for the brand data, because they are extremely expensive to obtain.
The final dataset, starting from the 10 fashion brands selected, is composed by 390.698 users and 13.344.030 posts. The set of users and posts are split to define the *community* of each brand. It is important to point out that the analysis needs all the posts

of the user in order to build its graph model: this is not possible for the users that set the privacy flag and the ones that do not share anything on the platform. So, for this work, the number of followers is not indicative of the dimension of the correspondent network, but it is necessary to consider the number of *active* users, the ones that share content on the platform in a public way. Table 5.1 shows the numerics for each community in the dataset.

Table 5.1: Community Numerics

| brand account | followers | private users | no-posts users | active users | community posts |
|---|---|---|---|---|---|
| muzungusisters | 66938 | 34576 | 3958 | 28404 | 2910277 |
| miguelinagambaccini | 83848 | 32186 | 12930 | 38732 | 670195 |
| zeusndione | 27624 | 13540 | 1694 | 12390 | 1170628 |
| dodobaror | 89500 | 40838 | 6527 | 42135 | 2974526 |
| lisamariefernandez | 83411 | 38743 | 5193 | 39475 | 3230855 |
| **athenaprocopiou** | **26591** | **12229** | **1434** | **12928** | **1301625** |
| **emporiosirenuse** | **3113** | **1374** | **0** | **1739** | **223376** |
| heidikleinswim | 27426 | 15330 | 0 | 12096 | 983286 |
| loupcharmant | 20446 | 4933 | 705 | 14808 | 1796469 |
| **daftcollectionofficial** | **3782** | **595** | **53** | **3134** | **458622** |

The highilighted rows are the communities analysed in detail: Emporio Le Sirenuse and Daft are the smallest, while Athena Procopiou is an example of a medium community. The dimension is defined by the column ”active users”, which is the number of followers that have shared at least one post and that are not private.

## 5.3 Classification

One experiment is related with a classification task, which is used to verify the effective power of the features extracted from the networks. In this test, the purpose is to discriminate between users that are *consumers*, and so possible target of the brand, and *non-consumers*, which is a macro-category that includes other brands, fashion bloggers and retailers among all.
The ground truth dataset is defined by domain experts, that manually classified a set of users, picked from the database as a sample of the domain, for a total of 472 labelled users. A network is obtained from these ground truth users, including all their posts, tags and user mentioned and it is referred to as **ground truth network**.
The algorithm selected for classification is the *random forest*, which is an ensemble method that allows for flexibility. It is kept fixed for all the tests, in order to effectively compare the different set of features exploited. Next sections describe in detail each considered set and then a final comparison is performed.

### 5.3.1 Social Classifier

The overall approach is to test the features derived from a network structure, but a **baseline** is needed in order to verify the validity of the process. For this reason, a set of features is defined for each user, using the account information: number of followers, number of following, ratio between followers and following, number of posts and a boolean value to check the presence of words in the biography, such as "shop", "store" and "e-commerce". These are usually referred in literature as *network features*, because they quantitatively represent the social networking of each user. A classifier is trained using these data, creating the baseline for comparison with the *real* network features, the ones referred to the entire graph structure around the user.

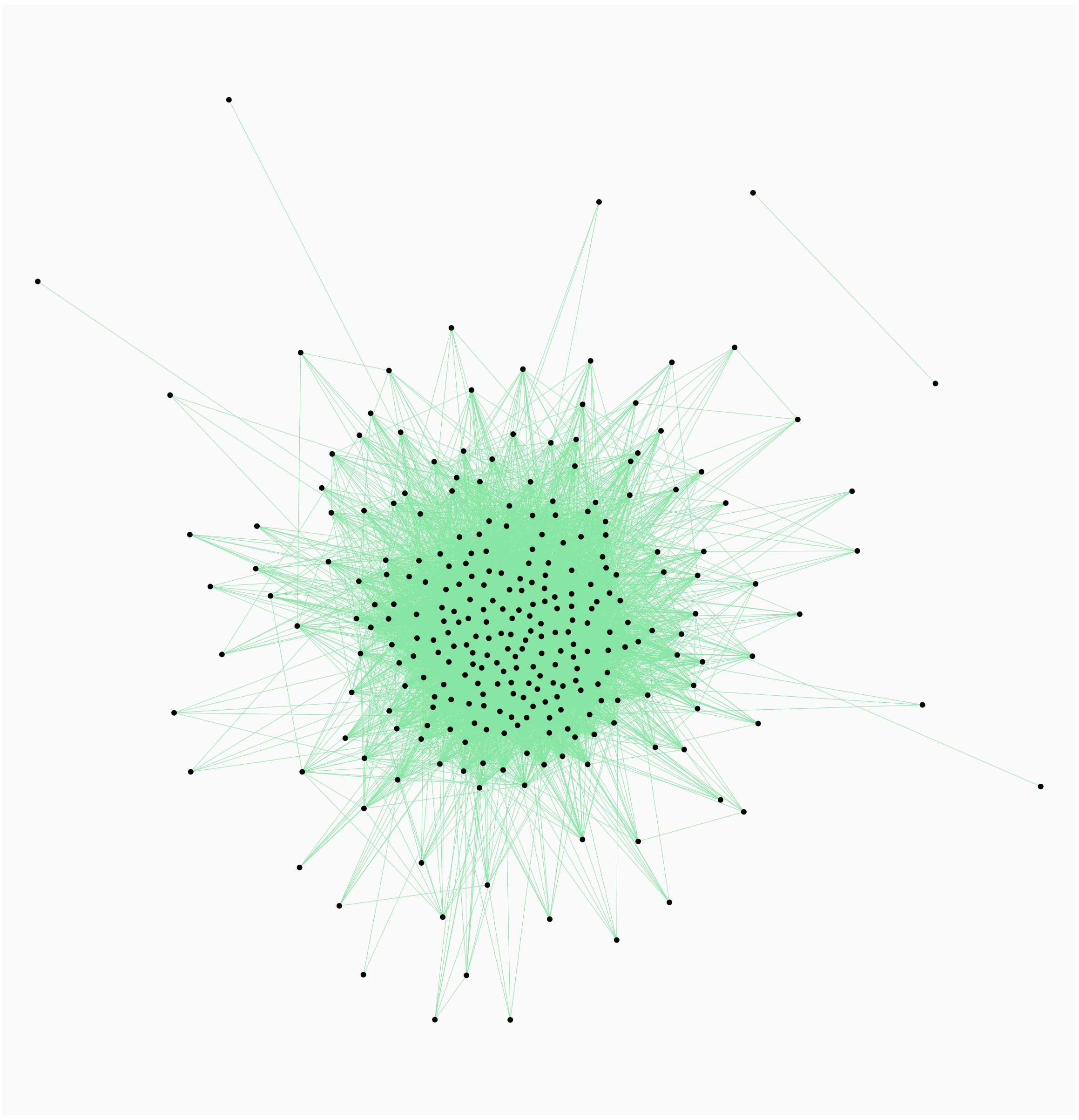
### 5.3.2 Complete Network Classifier

The ground truth network is used to extract the correspondent features. First, the mentions and hashtags networks are defined, as explained in the chapter 4, then, the embeddings are computed using the node2vec algorithm. A default parametrization is used and it is better explained in section 5.3.4. Finally, the features coming from the two networks are combined in order to generate the vector representative of each user and they are used as input for the classifier.
There is to notice that the complete networks are heterogeneous, so the features are computed also for the other types of node, but then these are discarded in the second part of the process. Moreover, number of nodes and edges is extremely high, which represents a computational problem during the embedding definition, which is extensively discussed in the last chapter.

### 5.3.3 Reduced Network Classifier

In parallel, with the method accurately defined in section 4.2.2, the reduced version of mentions and hashtags network is computed from the complete ones. As an example, the reduced network for hashtags is presented in figure 5.1.

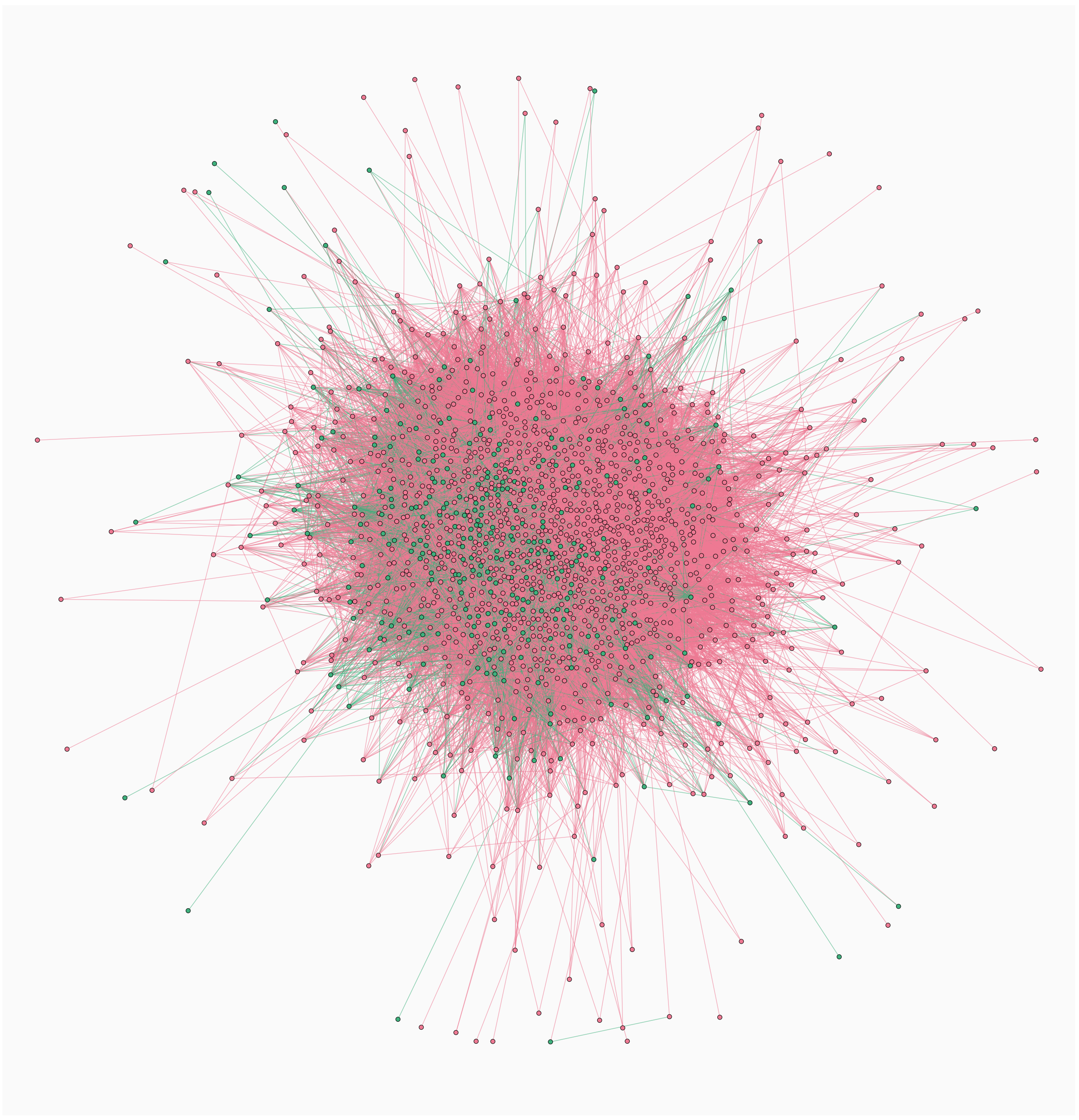
Figure 5.1: Ground Truth Reduced Network - Hashtags



The nodes represent the users and the edges connect each pair of user that shares at least one hashtag used in their posts. This network is a set of users coming from different communities inside the domain of the case study. It is possible to notice that two users are separated from the main graph, because they do not have anything in common with the rest of the users.

Thanks to the dramatic decrease in the number of edges and nodes for this kind of networks, the attention can be moved towards the *community networks*, which includes all the followers of a specific brand. These networks allow to exploit an intrinsic characteristics of the graph model: the neighbours of a labelled node, even if not labelled itself, contributes in the features definition. Non labelled nodes generate more common elements and, by consequence, an increased similarity between nodes in the features space. This specific behaviour, from a machine learning perspective, can be approached as **semi-supervised learning** and it allows to perform the classification using both few classified points and a lot of not classified points. This is exactly the setting in the case study considered, because only a small set of users is classified with respect to the entire community, since the ground truth is extracted cross-community. To give the idea of the contribution, the same graph of figure 5.1 is combined with the community network of Emporio Le Sirenuse in figure 5.2.
So, a further experiment is defined and the features of these extended networks are extracted in addition: the idea is that these ones have richer information, that can be a benefit for the classification task.

Figure 5.2: Ground Truth and Community Reduced Network - Hashtags



The green edges are the ground truth edges, the same as in figure 5.1, while the others (rose) are the not labelled ones, belonging to the Emporio Le Sirenuse community. It is clear that some of the nodes present in the ground truth network take advantage to the presence of other users: for example the two separated nodes present in the previous graph are connected in the lower part of this new one.

### 5.3.4 Hyper-Parameters Tuning

In classification, it is extremely important to correctly select the hyper-parameters of the model, a process also referred to as *model selection.* In the experiment, the node2vec algorithm has a lot of parameters that can be changed, increasing the flexibility but also the overall complexity. For this reason, in the pure classification step, the random forest classifier is defined with a default setting, without performing any optimization on its parameters: the focus is the *comparison* of the features, not the general top performance of the classifier.
For what concern the node2vec algorithm, needed to define the embeddings, the parameters are:

- return parameter $p$, already described in section 2.6.
- in-out parameter $q$, alredy described in section 2.6.
- number of dimensions $d$ for each node embedding.
- walk length $l$, the number of nodes sampled during each random walk.
- context size $k$, the maximum number of nodes to be considered in the neighbourhood.

It is clear that the perfect model selection is not feasible, so some assumptions are made: both $d$ and $l$ are chosen based on the dimension of the network, discriminating between complete networks ($d = 8$, $l = 40$) and reduced networks ($d = 4$, $l = 20$), while $k = 30$ is kept fixed in all the tests. The focus of the tuning is about the other two parameters, which are the most important, because they guide the type of search, thus defining the set of neighbours nodes.
After some preliminary trial, not reported for brevity, two main tests are developed: the first, which involves the set of social features, complete network features and reduced network features generated from the ground truth network. In this case, a standard parametrization is kept for both the complete and reduced features, the same parametrization used in the reference paper [11]. This is applied to detect, respectively, communities ($p = 1, q = 0.5$) and structural roles ($p = 1, q = 2$) and the two vectors obtained are merged to describe each user.
In the second test, thanks to the dimensionality reduction, *hyper-parameter grid search*, a standard methodology, is applied to select the best set of parameters. A range of values for $p$ and $q$ is defined, spanning from 0.2 to 5.0, then embeddings are computed for each pair $(p, q)$ and for each network type, they are combined and finally used to train the classifier. The final optimal results are based on 4 parameters $\hat{p}_{tag}$, $\hat{q}_{tag}$,$\hat{p}_{mention}$ and $\hat{q}_{mention}$, which are obtained by the combination of each possible pair. The top parameters obtained for each test are listed in table 5.3, while in table 5.2 the tests are briefly explained and then correlated with an id, that is used in all the following results for brevity.

Table 5.2: Tests Description

| id | features type | description |
|---|---|---|
| 1.1 | social | baseline, using quantitative features obtained from Instagram account |
| 1.2 | complete network | network structure features of ground truth *complete* network |
| 1.3 | reduced network | network structure features of ground truth *reduced* network |
| 2.1 | reduced network | network structure features of ground truth reduced network (hyper-parameters tuning) |
| 2.2 | reduced network | network structure features of ground truth reduced network merged with community of Emporio Le Sirenuse |
| 2.3 | reduced network | network structure features of ground truth reduced network merged with community of Daftcollection |
| 2.4 | reduced network | network structure features of ground truth reduced network merged with community of AthenaProcopiou |

In this table are presented the main test run, performed to exploit the network structure in different ways.

Table 5.3: Hyper-Parameters Tuning

| #test | $p_{tag}$ | $q_{tag}$ | $p_{mention}$ | $q_{mention}$ | accuracy |
|---|---|---|---|---|---|
| 2.1 | 3.4 | 3.4 | 3.4 | 3.4 | 0.637560 |
| 2.1 | 2.6 | 3.4 | 2.6 | 3.4 | 0.614448 |
| **2.1** | **0.2** | **0.2** | **5.0** | **3.4** | **0.673488** |
| 2.1 | 1.8 | 3.4 | 1.8 | 5.0 | 0.670113 |
| 2.2 | 5.0 | 1.0 | 5.0 | 1.0 | 0.671251 |
| 2.2 | 2.6 | 1.8 | 2.6 | 1.8 | 0.655266 |
| 2.2 | 2.6 | 1.8 | 0.2 | 1.8 | 0.716204 |
| **2.2** | **0.2** | **1.0** | **5.0** | **2.6** | **0.719166** |
| 2.3 | 5.0 | 4.2 | 5.0 | 4.2 | 0.669066 |
| 2.3 | 1.0 | 5.0 | 1.0 | 5.0 | 0.710285 |
| 2.3 | 1.8 | 5.0 | 5.0 | 2.6 | 0.707002 |
| **2.3** | **4.2** | **1.8** | **1.0** | **5.0** | **0.729114** |
| 2.4 | 1.8 | 0.2 | 1.8 | 0.2 | 0.709375 |
| 2.4 | 4.2 | 0.2 | 4.2 | 0.2 | 0.703125 |
| **2.4** | **4.2** | **0.2** | **1.8** | **0.2** | **0.743750** |
| 2.4 | 4.2 | 0.2 | 1.0 | 1.0 | 0.715625 |

The table presents a set of possible parametrization with the correspondent average accuracy: in bold there is the best result obtained for each test.

### 5.3.5 Evaluation

The classifiers are evaluated using *K-fold cross-validation*, with $K = 4$: the data is split in four chunks, three are used for training and the fourth for validation, then the process is repeated using another chunk as validation, until all the four are used. Finally the average accuracy is computed, where the accuracy is the fraction of correctly predicted samples with respect to the total.
The performance of the classifiers are presented in table 5.4 for the first test.

Table 5.4: Classification: First Test Results

| id | feature set | accuracy |
|---|---|---|
| 1.1 | social | 0.615440 |
| 1.2 | complete network | 0.536343 |
| 1.3 | reduced network | 0.641026 |

The results of the first test shows that the performance of the features extracted from the complete network is not satisfying, since the baseline is not outperformed. On the other side, the accuracy of the reduced version is quite interesting. Given this observation, the second test is developed, focusing on reduced networks, that allows to extensively study the hyper-parameters, as discussed in previous section. There is an important fact to point out: reduced networks focus on strong similarities between users, but they also eliminate all the users that do not show any similarity. In fact, if a user has no tag in common with any other user or if it does not use any tag at all, then it does not appear on the reduced hashtags network, while if it does not mention any user, then it does not appear in the reduced mentions network. This means that some users, that are present in the ground truth and that have a network representation, actually disappear in the correspondent reduced network. Combining the features help to include more users, because some users can show up only in one of the two networks: in this situation, the components of missing features are set to 0. On the other side, it is not possible to classify the users that do not appear neither in hashtags network, nor in mentions network. This drawback is reflected in the result table of the second test (table 5.5), as the indication of how many users are effectively classified.

Table 5.5: Classification: Final Results

| id | #users | network accuracy | baseline accuracy |
|---|---|---|---|
| 2.1 | 306 | 0.673488 | 0.598727 |
| 2.2 | 313 | 0.719166 | 0.649099 |
| 2.3 | 317 | 0.729114 | 0.634415 |
| 2.4 | 320 | 0.743750 | 0.646875 |

The ”network accuracy” column is referred to the best performance obtained in the test using network features, while the ”baseline accuracy” is the performance of the classifier that uses social features.

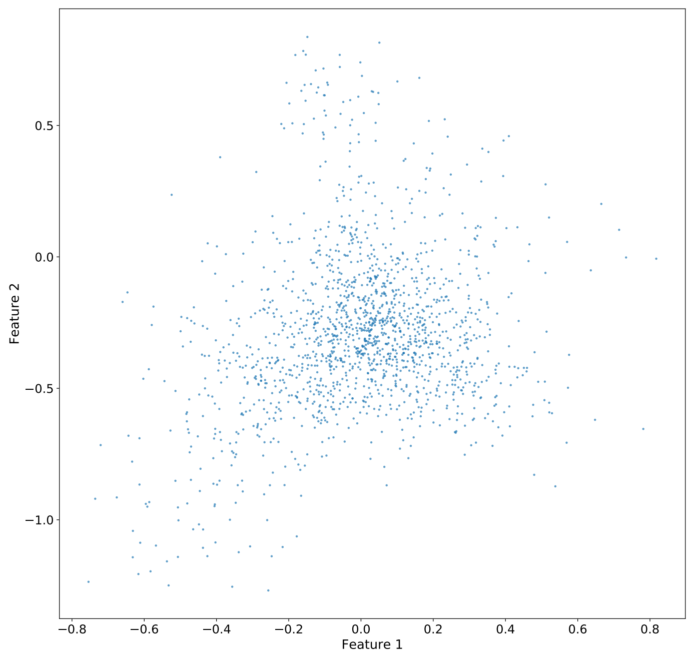
In order to have a correct comparison, first the network features are extracted, verifying what users are present, and then the social features are computed for the same subset of users. The classifier with network features overperform in all the cases the social one: the features are able to map very detailed information about the content and the relationships of each user, much more than the quantitative information present in the profile. Moreover, the inclusion of not labelled users increase both the number of users and the performances: having a set of users well connected allows to improve the feature vector itself and it introduces also stronger connections, that are used to include more users in the process. This is obtained, by the way, without any effort from the classification perspective, because the number of manually classified users is low, while the performance grows using a higher number of non labelled users.

## 5.4 Clustering

The other experiment defines a clustering task, that is strongly related with the classical community detection described in the previous chapters. The idea is to check whether there are groups of users, inside the macro-communities of followers, that form sub-communities and that are interesting for the brand. In fact, the intrinsic definition of clustering is that the groups defined ”naturally emerge from the data”, which means that can possibly show unknown but useful insights.
The process follows a similar pipeline with respect to classification, but in this case the features are always kept separated: there is no need to enrich the description of the users, because the aim is to capture a hidden common behaviour, which can be achieved if the focus is on a specific descriptive element of the data.
The steps defined are the standard ones in unsupervised learning and they are described in detail:

1. *Extract embeddings from the reduced network*
   There are four possible types of embeddings to be extracted, given by the combination of the two types of networks (hashtags and mentions) with the two parametrizations, that detect either communities or structural roles. In the next steps, only the embeddings generated by the *hashtags* network using the parameters to extract *communities* are reported (figure 5.3).

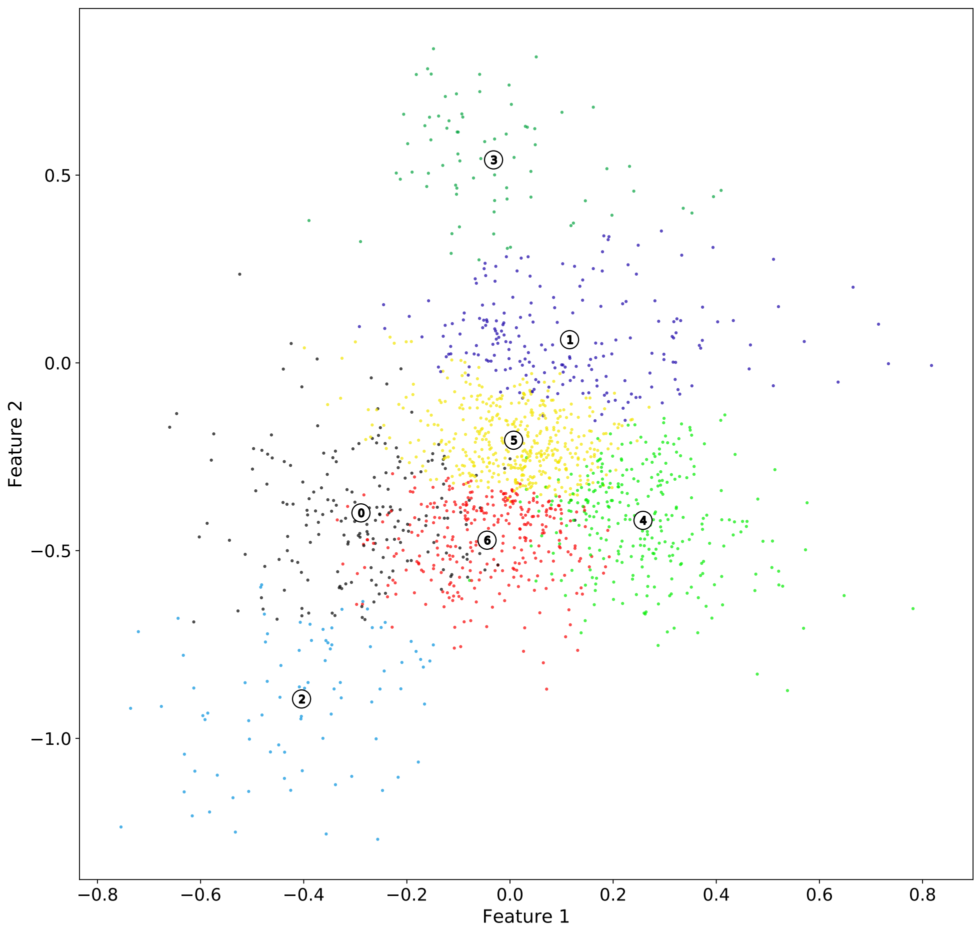
Figure 5.3: Embeddings Distribution



The distribution of points in this latent space is the mapping of the relative position of nodes inside the reduced network. The first 2 features components are used for visualization.

2. *Use embeddings as input for the K-means algorithm* (figure 5.4)

Figure 5.4: K-Means Output

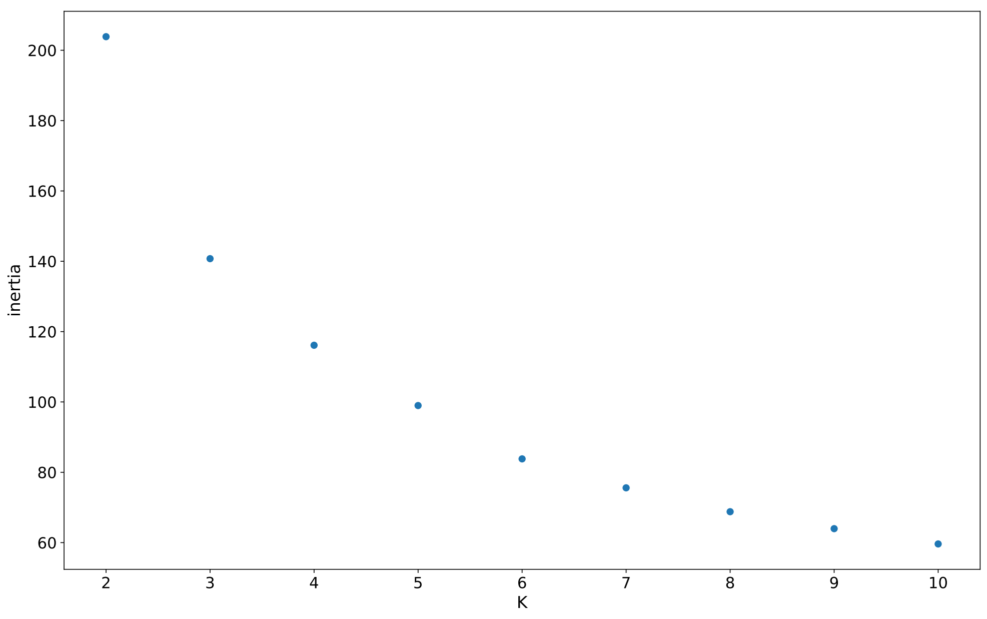


The figure shows the output of the K-means algorithm run with K $= 7$, the one validated with domain experts

3. *Verify cluster structure using internal measures*
   Inertia is the sum of squared distances between each point and the correspondent centroid, which means it is a decreasing function with the number of clusters by definition. It allows to detect the best K via the *elbow method*: plotting the number of clusters K against the inertia, the best value is obtained in correspondence of the ”elbow” in the chart. In this specific point, the sum of squared distance starts to stabilize and so it may be better to stop. In figure

5.5, the inertia is plotted against the number of clusters for the target data. There is no guarantee that is the best result and in some situation it is better to refer to external measures, too.

Figure 5.5: Inertia



The inertia is computed for different values of K for the dataset presented in figure 5.3: using the *elbow method* it is possible to check that a first net decrement happens at $K = 3$, then there is a stabilization after $K = 6$ and $K = 7$. The validation with domain experts, used as an external measure, points out that a higher $K$ value is needed for a more meaningful definition of the clusters.

As further structural analysis, another internal measure is used: the *silhouette coefficient*. The silhouette coefficient is a measure of how similar an object is to its own cluster (*cohesion*) compared to other clusters (*separation*). The silhouette ranges from $-1$ to $+1$, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighbouring clusters. If most objects have a high value, then the clustering configuration is appropriate, while if many points have a low or negative value, then the clustering configuration may have too many or too few clusters. The configurations are tested for various $K$ (figures 5.6, 5.7, 5.8, 5.9). In the left plot, the silhouette coefficient for each data point is showed, with the red line indicating the average value. In the right there is the visualization of the clusters using the first two features.

Figure 5.6: Silhouette Coefficient - K = 3

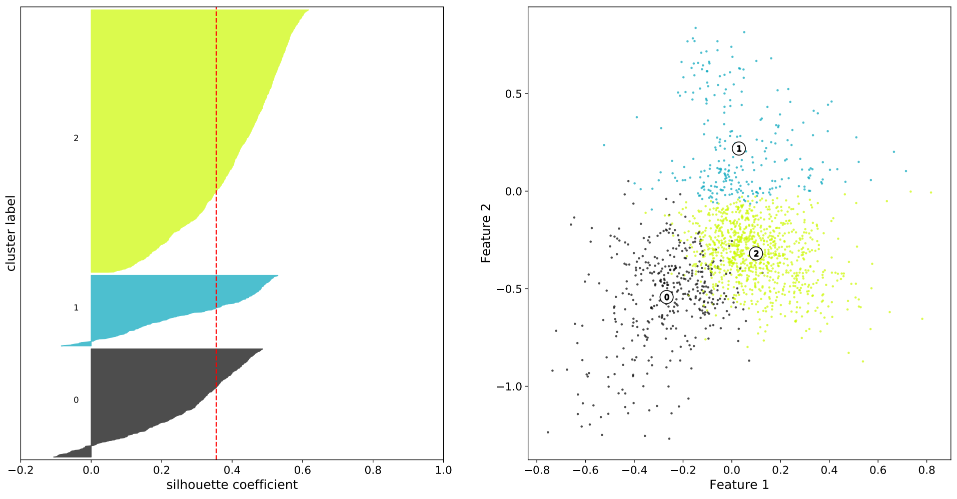


Figure 5.7: Silhouette Coefficient - K = 5

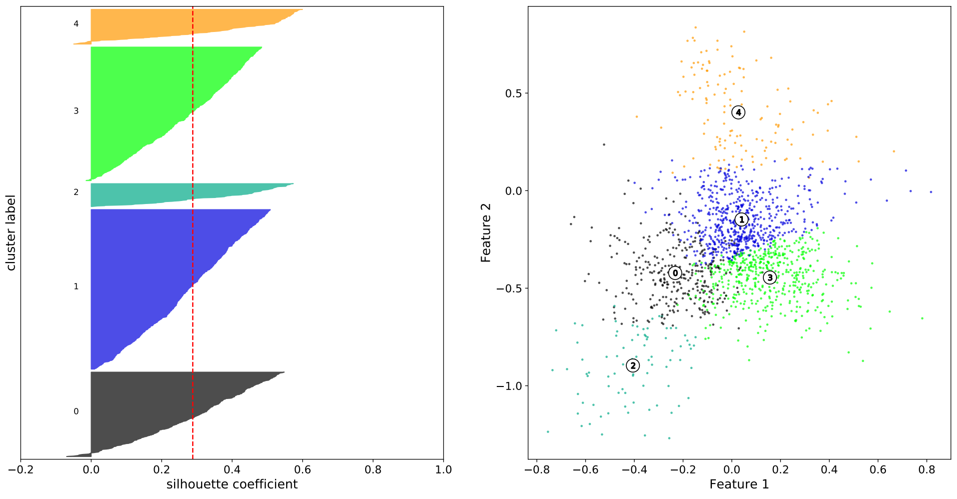


Figure 5.8: Silhouette Coefficient - K = 7

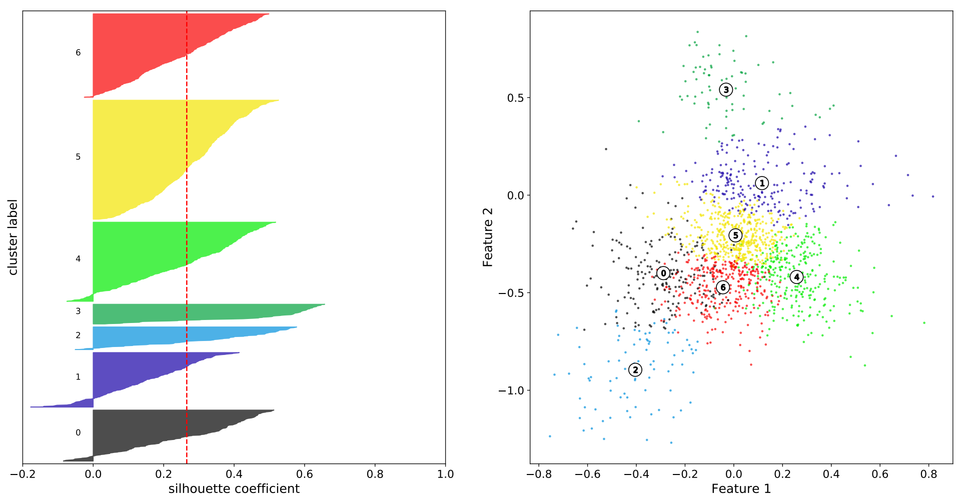
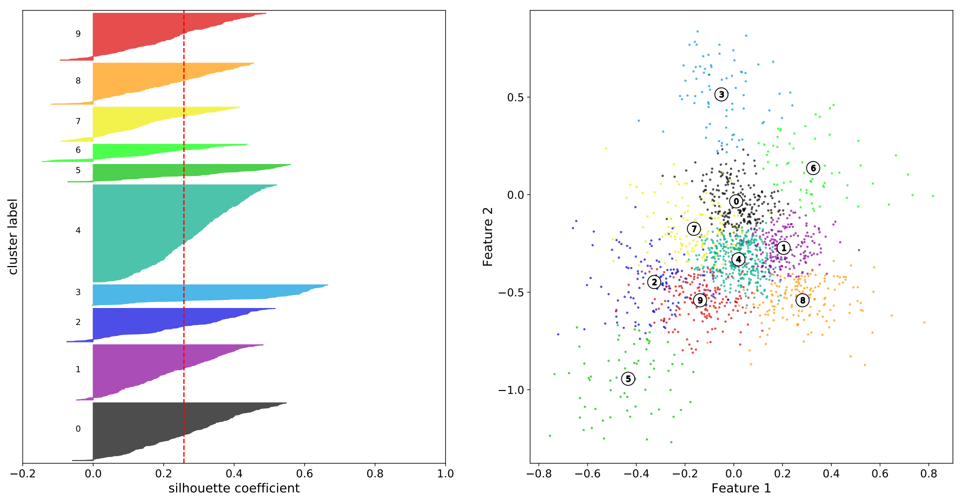


Figure 5.9: Silhouette Coefficient - K = 10



Also the silhouette coefficient does not give a unique interpretation about what is the best number of clusters.

4. *Validate the clusters*
   Validation is the hardest part, since it is an unsupervised method and ground truth data is not always available. Moreover, labelling for each group is needed in order to check the effectiveness of the method: in the case study, a support from domain experts is required in order to manually inspect the clusters obtained. A method to automatically extract labels and similarities from the data is developed to ease the validation.
   For what concern the similarity, the distance with respect to the centroid is computed and used to rank users inside the clusters. Then, the top 20 users are extracted for each cluster and used for validation. The label extraction, on the other hand, is based on how the reduced network is built: it connects users that share some elements, which are either other users mentioned or hashtags used in their posts. A cluster is no more than a subset of this graph (e.g.: a community) and the connections inside are based on the shared elements. Based on these observation, a possible way to extract a labelling is to compute the set of **common hashtags** for the cluster, in order to represent the entities that effectively increased the similarities of the users inside the clusters. Given $t_u$ and $t_v$ the sets of tags of users $u$ and $v$, the common tags can be defined as $T_{uv} = t_u \cap t_v$ and the common tags of a cluster $c$ are:

$$T(c) = \bigcup_{u,v \in c} T_{uv}$$

   This definition, along with the pruning of tags described in previous chapter, let only significant hashtags to be extracted, but the overall dimension depends on the number of users inside the cluster. Having a huge number of words to describe a cluster is very difficult to be effective, so an inverse process is applied: the set of users in the cluster is used to **re-expand** the network, in order to obtain the original heterogeneous network, zoomed around the users selected. In that way it is possible to easily compute the **most used hashtags** as the ones with highest in-degree, leading to the definition of the labels as the top 10 hashtags by frequency, that belong to $T(c)$, too. In table 5.6 are presented the set of tags extracted for each cluster of figure 5.4.

Table 5.6: Cluster Labels

| cluster | hashtags |
|---|---|
| 0 | madeinitaly, accessories, hat, handmade, luxury, bikini, bag, earrings, lifestyle, mensfashion |
| 1 | food, italianfood, whatitalyis, wedding, photo, delicious, foodlover, roma, summer, prettylittleitaly |
| 2 | interiordesign, design, decoracao, interiors, interior, chic, inspiration, art, designer, architecturephotography |
| 3 | amalficoast, positano, capri, praiano, villatreville, AmalfiCoast, isleofcapri, browsingitaly, sugokuiievents, costieramalfitana |
| 4 | nyc, blessed, london, foodie, family, baby, latergram, nyfw, yum, biarritz |
| 5 | malibu, life, wedding, greece, goodvibes, wanderlust, travelgram, photography, hipstamatic, instaday |
| 6 | architecture, vintage, happiness, art, ashowroom, beauty, ashowroominparis, illustration, botanicalbeauty, followthebuyers |

The table shows an example of labels extracted for each cluster. Some of these are extremely specific, thanks to the pre-processing phase, in which the stop words of Instagram platform are eliminated.

The same methodology can be reproduced with **common mentioned users** for the mentions network in the same exact way, even if the results are more difficult to interpret: the label would be composed by a set of users. Due to this fact, the clustering effectively validated with domain experts is the one referred to sub-communities extracted by the reduced hashtags network of Emporio Le Sirenuse community.
The experts positively rated both the clusters 0, 1, 2, 3 and their correspondent labels: the users inside this clusters are effectively similar, sharing similar content, and the labels match the kind of content. On the other side, the remaining clusters are not well defined: users inside these groups post a variety of different things and it is difficult to verify if there is any similarity. This fact depends of profile's types: in fact, the first three clusters are professionals and brands, which want to characterize themselves on the social media in a very specific manner and so are easier to be modelled. Instead, the last three groups are composed by "regular" users and so a pattern is more difficult to be highlighted, because the variety of content increases. There is to point out that cluster 3, instead, is composed by normal users that are extremely connected, posting pictures of sea and beaches, sharing the idea of vacation. Given this feedback, the conclusion is that the process developed, combined with the clustering, is a very good method in order to extract groups of professionals, while further analysis need to be run for regular users, because only in one case the match is correct and identifiable.

As final result, the network presented at the beginning (5.2) is visualized using the clusters validated in figure 5.10.

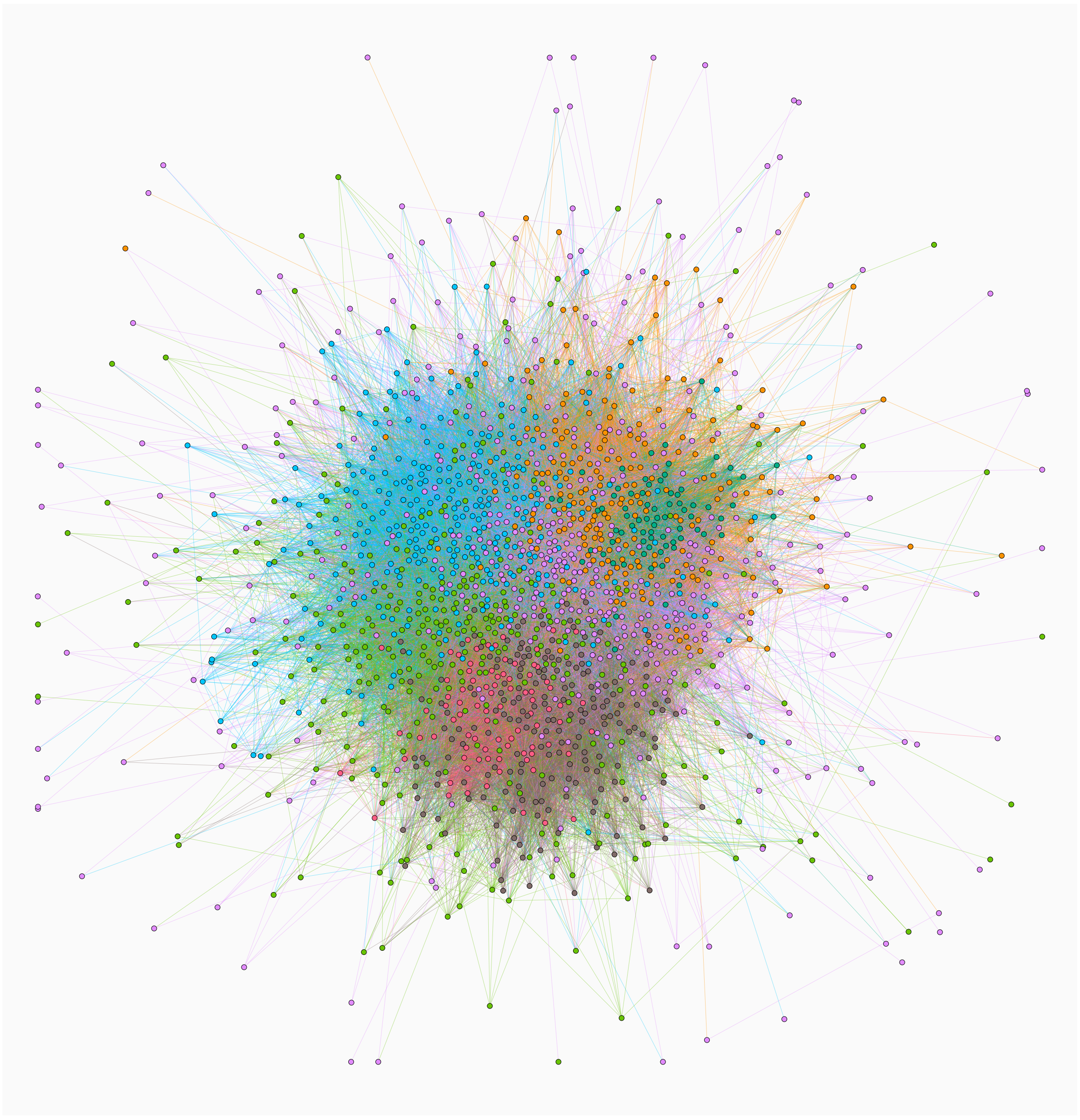


Figure 5.10: Community Reduced Network - Clusters

The figure shows the network using the seven clusters obtained in this section.

## 5.5 Comparison: Classification and Clustering

The methods presented in the two previous sections are applied to the same set of users, represented with different features extracted from networks but with the same general objective: give a description of these users and, possibly, group together the ones that are most similar.
The idea of comparison arises from the observation that some clusters are more representative of other brands, professionals and showrooms, which are basically **non consumer** users. This means that clustering output can match the labels of the ground truth defined in section 5.3, if K is set to 2. In order to verify this possible overlap, the K-means algorithm is run with $K = 2$ using the same dataset and features represented in figure 5.3 and the correspondent outputs are compared with the class labels. The resulting comparison is presented in table 5.7.

Table 5.7: Cluster comparison with ground truth - K = 2

| | consumer | non consumer |
|---|---|---|
| **cluster 0** | 135 (75.84 %) | 43 (24.16 %) |
| **cluster 1** | 46 (38.33 %) | 74 (61.67 %) |

The percentage is referred to the total number of users in the cluster: cluster 0 can be correspondent to class *consumer*, while cluster 1 can be labelled as *non consumer*. If this output is considered as the prediction of each class, the obtained accuracy is around 0.70, which is lower than the one obtained using the feature combination developed in section 5.3.

This comparison shows that clustering is able to identify with a good level of accuracy, around 70%, the consumer and non consumer users, but the classifiers implemented in section 5.3 perform better. On the other hand, if this result is compared with the clustering of section 5.4, other interesting insights can be made. In fact, the seven clusters can be split into two super-groups: the ones more related with professionals and brands and the ones with regular users. This means that, increasing the number of clusters, not only the high-level categorization is still present, but also sub-categories are retrieved: for instance, cluster 3 is composed by regular users that enjoy holidays and the sea, while cluster 2 contains professional users belonging to interiors design domain. The conclusion is that *increasing* the number of clusters leads to extract specific sub-communities that either belong to the consumer or non-consumer domain. Using the validated clusters, with $K = 7$, the distribution of labelled users verifies this observation: the resulting overlaps are presented in table 5.8.

Table 5.8: Cluster comparison with ground truth - K = 7

| | consumer | non consumer |
|---|---|---|
| **cluster 0** | 10 (24.39 %) | 31 (75.61 %) |
| **cluster 1** | 8 (72.73 %) | 3 (27.27 %) |
| **cluster 2** | 7 (46.67 %) | 8 (53.33 %) |
| **cluster 3** | 1 (50.00 %) | 1 (50.00 %) |
| **cluster 4** | 51 (72.86 %) | 19 (27.14 %) |
| **cluster 5** | 49 (73.13 %) | 18 (26.87 %) |
| **cluster 6** | 55 (59.78 %) | 37 (40.22 %) |

Cluster percentages show that for 5 clusters the manual validation matches with the labelling: clusters 0 and 2 have mostly non consumer users, while in 4, 5 and 6 predominate the consumers. Nothing can be said for cluster 3, since there is an equally distribution of the only two users included.

This hypothesis is only partially supported by the results because the numerics are not high enough for each group. Other studies are needed to have more solid confirmation: in case of success, these would be of extremely high value for the brand, because it would have a way to automatically deepen the knowledge of the hidden communities around the brand itself.

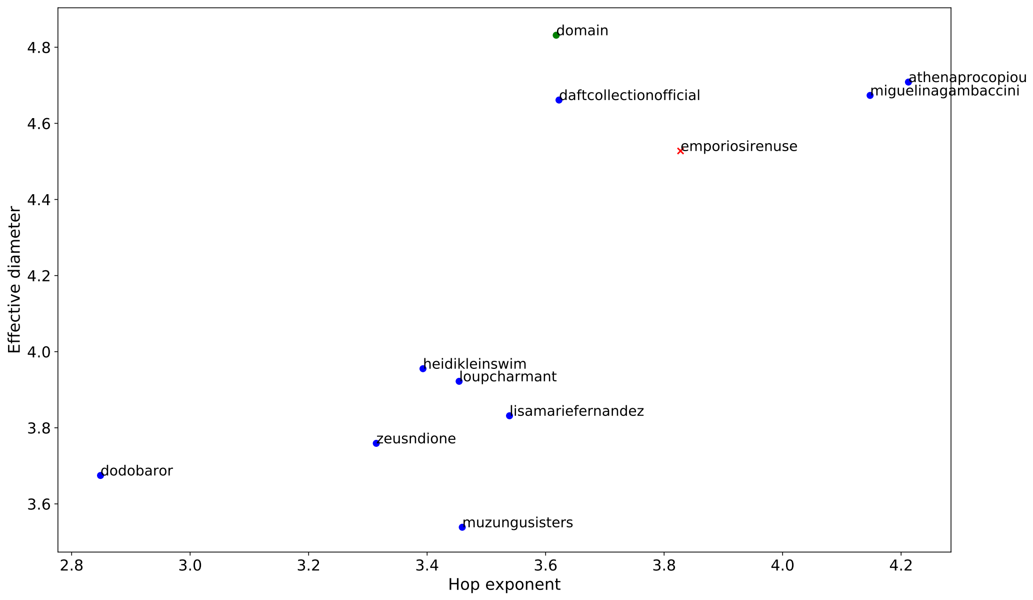
## 5.6 Other Results

In this final section, results of minor analysis are presented: they are techniques applied to the brand networks instead of the community ones, so most of the insights are much different and cannot be generalized, while some are part of the process that has led the thesis on community network analysis.

### 5.6.1 Brand Networks

Brand networks are characterized by more details, because also comments and geolocation are gathered, but with less volumes: the overall number of posts for each brand is very low, with an average value of 213 posts, which are distributed in 9 months. The focus of the first analysis is based exactly only on the pure network structure generated by the brands, in order to compare their dimensions, giving a general insight on the social activity. In figure 5.11, brands are represented by structural dimensions and plotted in this space, to check the high level similarities present. Unfortunately, the results are not very significant, since the network of posts is quite simple and the values are not so discriminating.

Figure 5.11: Brands Network Structure

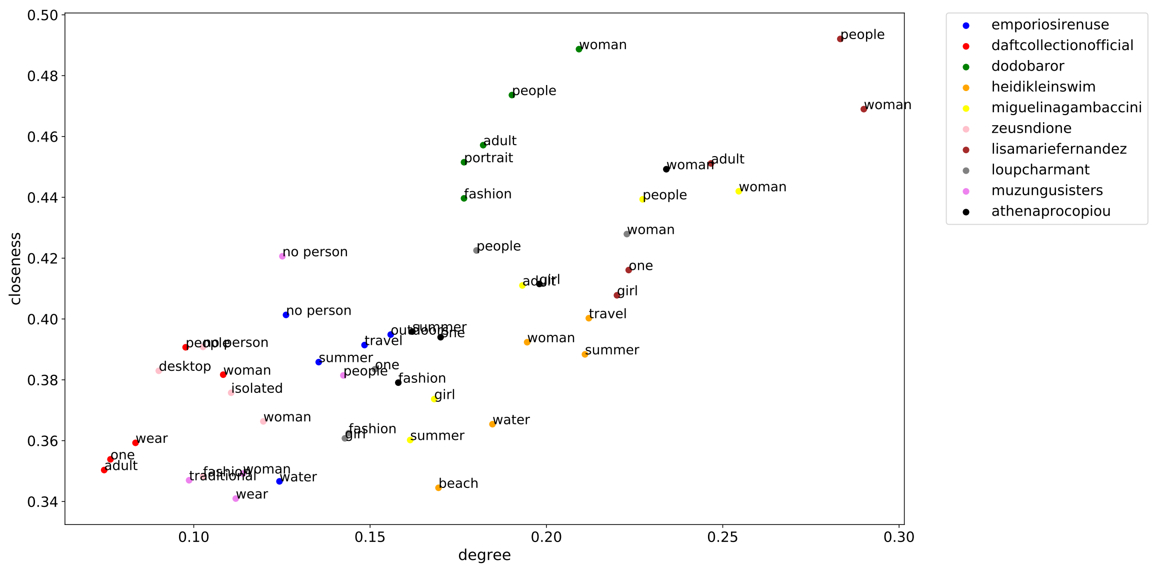


The brand structure is described in function of the effective diameter, the 90th percentile of the shortest path length distribution of the graph, and the hop exponent, the exponents that approximate a graph as a power law distribution

## 5.6.2 Image Concepts Networks

An interesting result is obtained comparing the general concepts produced by the brands: it is clear from figure 5.12 that the brands have some common ideas and that they want to share them in a very clear way. For instance concepts as *woman* and *people* are the top for almost all the brands, while only some brands put the accent on *summer* or *travel*.
This kind of analysis would be extremely important if it was reproduced for the entire community of users and if the tags extracted was effectively significant in the fashion domain. The first problem is mostly computational: the Clarifay API have limitations and processing hundred of thousands of images would require a lot of time. As for the second problem, the models implemented in the API are quite generic and so a more specific model is required in order to obtain interesting evidences.

Figure 5.12: Brands Top Image Concepts

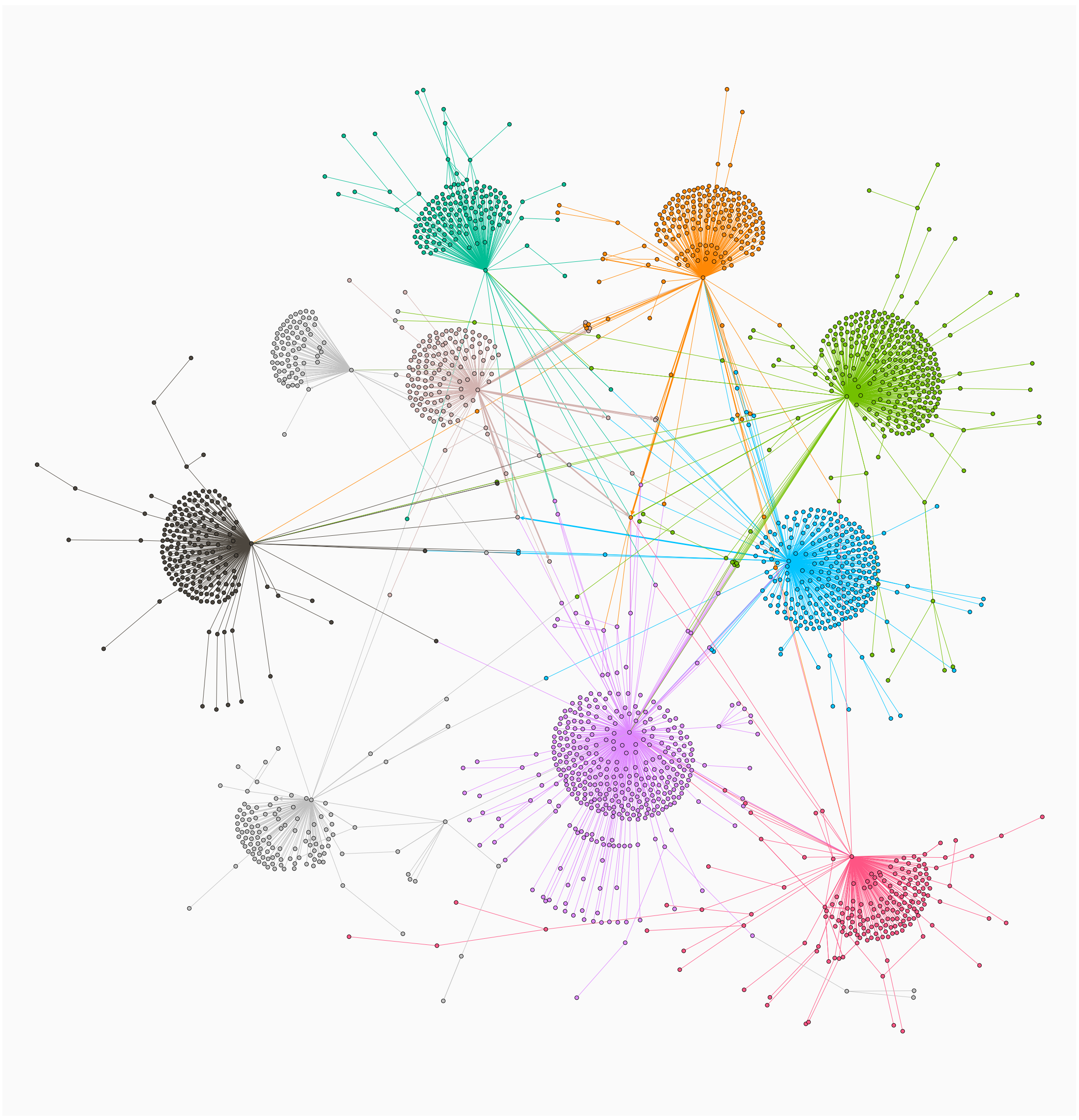


The normalized version of centrality measures allow to compare the importance of image concepts belonging to the networks of each brand. The degree centrality (x-axis) represents the frequency of the concept with respect to the total, while the closeness (y-axis) defines the most shared ones, because a high value means that the other nodes in the network can be easily reached.

## 5.6.3 Mentions Networks

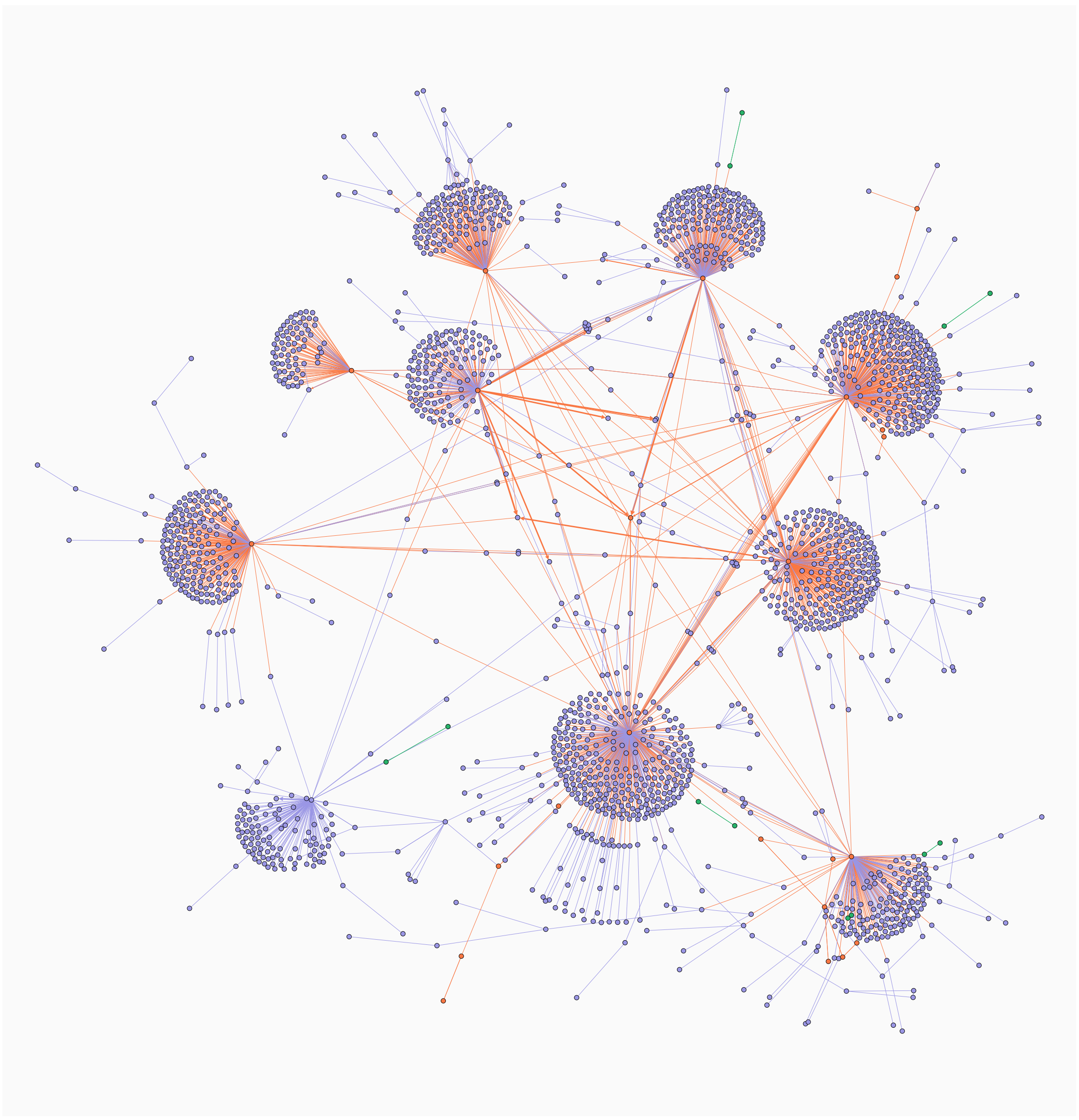
The most important result in this section is given by the mentions network: the presence of other users makes the analysis much more interesting, because connection points can be highlighted, as long as roles and communities. In fact, this is the same technique developed for community networks, applying K-means to the features generated by the standard parametrizations: in figure 5.13 the communities are highlighted, while in 5.14 the structural roles.

Figure 5.13: Brands Mentions Network - Communities



The communities are easily detectable, even visually, because the mentions define a very specific relationship between users, expecially for brands: they use this feature to highlight partnership and collaborations.

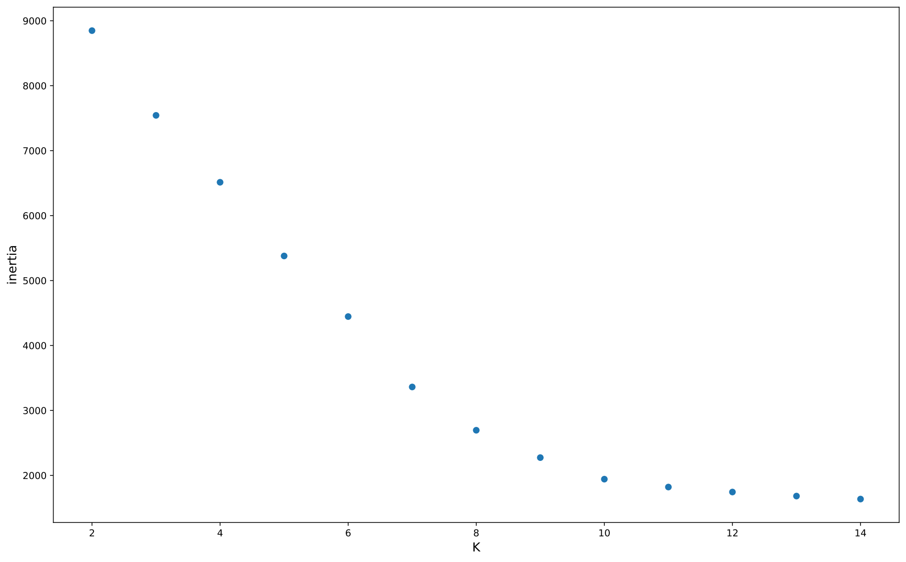
Figure 5.14: Brands Mentions Network - Structural Roles



There are two main roles: the *brand role* (orange) and the *mentioned role* (blue). Even if the data is built around brands and this separation is automatical, it is interesting to point out that there are some nodes that are assigned the same role as brands, bringing them more attention in the study.

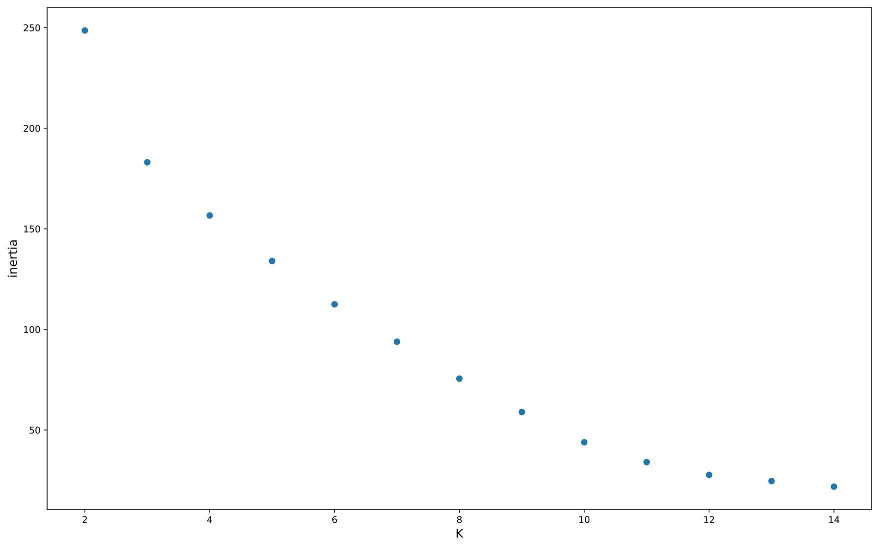
As for K selection, in this setting it is extremely evident using an internal measure: in figure 5.15 and 5.16 the inertia for the community detection and for the role detection are presented.

Figure 5.15: Inertia - Communities



The ”elbow” is in correspondence of $K = 10$, which is the expected number of communities.

Figure 5.16: Inertia - Structural Roles



In this case the identification of the best K is not so evident, but forcing more than 3 roles defines extremely small clusters, which become not relevant.

It is important to underline that the nodes in the middle of the graph, shared among more brands, have a sharply defined cluster: edges weights let the nodes to be positioned in the latent space much nearer to the brand that makes more mentions and so the cluster definition is almost unique. For a better understanding, table 5.9 shows the overlap of these nodes "in the middle", considering the overlap as the ratio between the mentions of a brand with respect to the overall mentions received.

Table 5.9: Users shared between communities

| user | community | overlap |
|---|---|---|
| amberfillerup | miguelinagambaccini | 0.214286 |
| **amberfillerup** | **athenaprocopiou** | **0.785714** |
| sorayabakhtiar | miguelinagambaccini | 0.142857 |
| **sorayabakhtiar** | **athenaprocopiou** | **0.857143** |
| matchesfashion | muzungusisters | 0.006711 |
| matchesfashion | zeusndione | 0.040268 |
| matchesfashion | dodobaror | 0.241611 |
| matchesfashion | lisamariefernandez | 0.147651 |
| **matchesfashion** | **athenaprocopiou** | **0.268456** |
| matchesfashion | loupcharmant | 0.147651 |
| matchesfashion | daftcollectionofficial | 0.127517 |

The table shows example of users that are mentioned by more than one brand, where overlap is the ratio of the mentions made by one brand divided by the overall mentions received: "matchesfashion" is the only example that is effectively shared among more brands and in fact, in the network it is the node in the centre. Even with these values, the clustering achieves to put it in the correct community (athenaprocopiou), since the number of mentions is used in the embeddings definition.

Given these results, the same approach is applied also to other networks, such as the comments networks and the likes networks. Since likes and comments are collected only for the brands accounts, as pointed out in section 5.2, the network information for these users is not defined. For this reason, the models built using this data are extremely incomplete and a suitable analysis cannot be performed.

# Chapter 6

# Conclusions

The work of thesis presented deals with a very powerful model, the graph model, and it exploits the most recent techniques in order to extract useful information from on-line social networks. In general, the network approach is applied extensively to model many different aspects of the users present in the case study, first focusing on the brands and then on their communities. A set of techniques is applied to two macro-categories of networks: brand networks and community networks. In the specific, the focus is on the definition of a method to handle in an efficient way the heterogeneity of the social network structure, in order to encode all the data needed in a simpler graph model: the method presented allows to extract a classical social network, with only user nodes, from a much more complex network, without losing the necessary information to effectively capture user behaviour. A very recent approach, representation learning, is applied to these networks in order to describe user nodes in term of a continuous feature vector, that is used to perform classification and clustering. The classification task confirms the power of this features, overperforming the social baseline in all the tests defined: the encoded network structure is able to capture social behaviours of users more than quantitative features themselves. The experiment highlights also the importance in the process of not labelled users: they enrich the graph with more valuable relationships, that can be exploited to build more meaningful vectors, defining better similarities. The clustering task, on the other hand, allows to extend the classical community detection problem, introducing flexibility in the number of communities detected along with content validity. Most of the groups extracted contain users that are very similar, which is verified with domain experts. Also the method can be refined in order to extract subgroups more specific, putting the accent on possible sub-categorizations.
The work done follows standard and verified procedures but some limitations and drawbacks are encountered. These are presented together with possible solutions and future works:

- *Computational Limits*
  The generation of the basic networks using SNAP is very efficient, but the step of embedding generation using the node2vec algorithm is extremely time and memory consuming. For this reason, the tests are performed only on the

three smallest communities where all the steps can be performed with the available resources. Actually, the generation of a single set of embeddings is only memory-consuming, but time becomes critical during the hyper-parameters search, where a set of different embeddings needs to be tested to find the optimal configuration. A possible extension, in this direction, is to study the already optimized networks in order to discover what is the best underlying model to represent them and to extend this optimal parametrization to bigger networks. In this way, this costly step would be skipped, without losing the validity of the process and including in the analysis the biggest communities.

- *Process Limits*
  The most important drawback of the overall process is the loss of users in the reduction of the network, because no strong relationship can be obtained from the data: for this set of users it is impossible to perform any kind of analysis, since they are not present in the networks. On the other hand, there is to point out that, if there is no strong relationship, nothing valid can be extracted for these users. A possible improvement would be to encode other kind of networks, such as the image concepts network, so to enrich the description of each user, possibly mapping new similarities.

- *Information Limits*
  The approach used does not take into account the *time* variable: time is a key feature that could help to model more specific behaviours. In this work, some preliminary tests have been defined, consider a set of possible techniques to encode time into dynamic networks, but none of them has had interesting results. More studies can be made in this direction, trying to verify what are the interesting time patterns on social media and what kind of time windows can be considered in order to to extend the presented method with the dynamic component.

# Bibliography

[1] Leman Akoglu, Hanghang Tong, and Danai Koutra. Graph based anomaly detection and description: a survey. *Data Mining and Knowledge Discovery*, 29(3):626–688, 2015.

[2] Mohammad Al Hasan and Mohammed J Zaki. A survey of link prediction in social networks. In *Social network data analytics*, pages 243–275. Springer, 2011.

[3] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

[4] Ulrik Brandes, Rudolf Kruse, and Myra Spiliopoulou. *Community analysis in dynamic social networks.* PhD thesis, Ph. D. dissertation, angenommen durch die Fakultat fur Informatik der Otto-von-Guericke-Universit, 2009.

[5] Shiyu Chang, Wei Han, Jiliang Tang, Guo-Jun Qi, Charu C Aggarwal, and Thomas S Huang. Heterogeneous network embedding via deep architectures. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 119–128. ACM, 2015.

[6] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.

[7] Rion Brattig Correia, Lang Li, and Luis M Rocha. Monitoring potential drug interactions and reactions via network analysis of instagram user timelines. In *Biocomputing 2016: Proceedings of the Pacific Symposium*, pages 492–503. World Scientific, 2016.

[8] Linton Freeman. The development of social network analysis. *A Study in the Sociology of Science*, 1, 2004.

[9] Linton C Freeman. Centrality in social networks conceptual clarification. *Social networks*, 1(3):215–239, 1978.

[10] Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826, 2002.

[11] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[12] Lin Gui, Yu Zhou, Ruifeng Xu, Yulan He, and Qin Lu. Learning representations from heterogeneous network for sentiment classification of product reviews. *Knowledge-Based Systems*, 124:34–45, 2017.

[13] David Hallac, Youngsuk Park, Stephen Boyd, and Jure Leskovec. Network inference via the time-varying graphical lasso. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 205–213. ACM, 2017.

[14] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, pages 1025–1035, 2017.

[15] William L Hamilton, Rex Ying, and Jure Leskovec. Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584*, 2017.

[16] Kun He, Yingru Li, Sucheta Soundarajan, and John E Hopcroft. Hidden community detection in social networks. *Information Sciences*, 425:92–106, 2018.

[17] Mingqing Huang, Guobing Zou, Bofeng Zhang, Yue Liu, Yajun Gu, and Keyuan Jiang. Overlapping community detection in heterogeneous social networks via the user model. *Information Sciences*, 432:164–184, 2018.

[18] Instagram. www.instagram.com, 2017.

[19] Sergey Korolev and Leonid Zhukov. Supervised learning for link prediction using similarity indices.

[20] Jure Leskovec and R Sosič. Snap: Stanford network analysis platform, 2013.

[21] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *journal of the Association for Information Science and Technology*, 58(7):1019–1031, 2007.

[22] Haiying Liu, Lifang Wu, Dai Zhang, Meng Jian, and Xiuzhen Zhang. Multi-perspective user2vec: Exploiting re-pin activity for user representation learning in content curation social network. *Signal Processing*, 142:450–456, 2018.

[23] Alan Mislove, Massimiliano Marcon, Krishna P Gummadi, Peter Druschel, and Bobby Bhattacharjee. Measurement and analysis of online social networks. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 29–42. ACM, 2007.

[24] Seyed Ahmad Moosavi, Mehrdad Jalali, Negin Misaghian, Shahaboddin Shamshirband, and Mohammad Hossein Anisi. Community detection in social networks using user frequent pattern mining. *Knowledge and Information Systems*, 51(1):159–186, 2017.

[25] Mark EJ Newman. Fast algorithm for detecting community structure in networks. *Physical review E*, 69(6):066133, 2004.

[26] Mark EJ Newman. Modularity and community structure in networks. *Proceedings of the national academy of sciences*, 103(23):8577–8582, 2006.

[27] Evelien Otte and Ronald Rousseau. Social network analysis: a powerful strategy, also for the information sciences. *Journal of information Science*, 28(6):441–453, 2002.

[28] Sheng Pang and Changjia Chen. Community analysis of social network in mmog. *International Journal of Communications, Network and System Sciences*, 3(2):133, 2010.

[29] Marco Pennacchiotti and Ana-Maria Popescu. A machine learning approach to twitter user classification. *Icwsm*, 11(1):281–288, 2011.

[30] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. *arXiv preprint arXiv:1703.06103*, 2017.

[31] Carla Sersale. Emporio le sirenuse, 2017.

[32] Fatemeh Sheikholeslami and Georgios G Giannakis. Robust overlapping community detection via constrained egonet tensor decomposition. 2018.

[33] Chuan Shi, Yitong Li, Jiawei Zhang, Yizhou Sun, and S Yu Philip. A survey of heterogeneous information network analysis. *IEEE Transactions on Knowledge and Data Engineering*, 29(1):17–37, 2017.

[34] SNAP. Snap.py: Snap python interface, 2017.

[35] Karsten Steinhaeuser and Nitesh V Chawla. Community detection in a large real-world social network. In *Social computing, behavioral modeling, and prediction*, pages 168–175. Springer, 2008.

[36] Alex Hai Wang. Detecting spam bots in online social networking sites: a machine learning approach. In *IFIP Annual Conference on Data and Applications Security and Privacy*, pages 335–342. Springer, 2010.

[37] Zaher Yamak, Julien Saunier, and Laurent Vercouter. Sockscatch: Automatic detection and grouping of sockpuppets in social media. *Knowledge-Based Systems*, 2018.

[38] Xianqi Yu, Yuqing Sun, Elisa Bertino, and Xin Li. Modeling user intrinsic characteristic on social media for identity linkage. In *Proceedings of the 2018 ACM Conference on Supporting Groupwork*, pages 39–50. ACM, 2018.

[39] Matthew Zeiler. Clarifai: visual recognition api, 2017.

[40] Yadong Zhou, Dae Wook Kim, Junjie Zhang, Lili Liu, Huan Jin, Hongbo Jin, and Ting Liu. Proguard: Detecting malicious accounts in social-network-based online promotions. *IEEE Access*, 5:1990–1999, 2017.