UPPSALA
UNIVERSITET

UPTEC IT 17 015

Examensarbete 30 hp
Augusti 2017

Interpretable, Probabilistic Divisive Clustering of Large Node- Attributed Networks

Adam Ruul

UPPSALA
UNIVERSITET

**Teknisk- naturvetenskaplig fakultet
UTH-enheten**

Besöksadress:
Ängströmlaboratoriet
Lägerhyddsvägen 1
Hus 4, Plan 0

Postadress:
Box 536
751 21 Uppsala

Telefon:
018 – 471 30 03

Telefax:
018 – 471 30 00

Hemsida:
<http://www.teknat.uu.se/student>

Abstract

Interpretable, Probabilistic Divisive Clustering of Large Node-Attributed Networks

Adam Ruul

Clustering of social networks, known as community detection is a fundamental part of social network analysis. A community (also known as a module or cluster) is a set of nodes grouped together according to some characteristic. Traditionally, a community has been thought of as a set of nodes that are more densely connected with each other than the rest of the network. Introducing node attributes to a social network, allows for two possible sources of information when clustering the network: The network structure, and the attributes describing the nodes. Traditional community detection methods supporting both these sources of information tend to be computationally complex and the resulting clusters are difficult to interpret in the sense of what characteristic they were grouped on. We present two methods (probabilistic divisive clustering and top-sampled community search) built on-top of already existing community detection methods (CESNA and FocusCO). Both of our methods aim to detect communities with a specified attribute association, yielding interpretable results in a feasible amount of time. The community detection algorithms our methods are built upon, are applied to different datasets in order to examine the runtime performance. We also display how our proposed methods can be used to detect communities formed around topics of interest and how a network can iteratively be clustered in order to detect sub-communities with a specified attribute association. In conjunction with research about psychological profiling, we believe that our proposed methods could be used to detect communities of people having similar psychological profile in online social networks.

Handledare: Lisa Kaati
Ämnesgranskare: Michael Ashcroft
Examinator: Lars-Åke Nordén
ISSN: 1401-5749, UPTec IT 17 015
Tryckt av: Reprocentralen ITC

Populärvetenskaplig sammanfattning

Klusteranalys tillämpad på sociala nätverk, känt som *community detection* är en fundamental del av social nätverksanalys. En community, (även känt som gruppering, modul eller kluster) är en uppsättning av personer (representerade som noder), grupperade efter gemensamma egenskaper. Traditionellt sett har en community ansetts vara en uppsättning av noder som är mer tätt sammankopplade med varandra, än med resten av nätverket. När nodattribut introduceras i ett socialt nätverk, finns det två informationskällor som kan användas när man grupperar nätverkets noder i olika communities. Den första informationskällan är nätverksstrukturen, den andra är attributen som beskriver varje nod. Traditionella metoder för community detection som tar hänsyn till båda dessa källor är ofta beräkningsmässigt komplexa. Dessutom genererar dessa metoder kluster som är svårtolkade i den mening att det är svårt att se vilka egenskaper noderna har gemensamt. Vi presenterar två nya metoder, baserat på två redan existerande metoder CESNA och FocusCO. Syftet med våra två metoder är att identifiera communities med en specifierad uppsättning av attribut associationer och generera lättolkade resultat under en rimlig tidsram. De två metoder som våra metoder är baserade på, testas flera datauppsättningar med syftet att undersöka skalbarheten för dessa metoder. Vi visar även hur våra framtagna metoder kan användas för att identifiera communities som har formats kring olika intressen samt hur det sociala nätverket kan iterativt klustras för att identifiera sub-communities med en specifik uppsättning av attributassociationer. Tillsammans med forskning kring psykologisk profilering och *psychometrics*, tror vi att våra framtagna metoder kan användas i sociala nätverk för att identifiera communities av personer med liknande psykologiska drag.

Contents

1	Introduction	1
1.1	Aim & objectives	1
1.1.1	Delimitations	2
1.2	Overview of this thesis	2
1.2.1	Background	3
1.2.2	Our Contribution	3
1.2.3	Psychographic clustering of social networks	3
2	Related work	5
3	Theory	6
3.1	Social network analysis	6
3.1.1	Community detection	6
3.2	Community detection with node-attributed graphs .	8
3.2.1	Node-attributed networks	8
3.3	Community detection in large scale networks	10
3.4	Community detection vs. community search	11
3.5	CODICIL	12
3.6	CESNA	15
3.7	FocusCO	18
3.7.1	Algorithm details	19
3.8	Ethical aspects	21
4	Probabilistic divisive clustering of node-attributed networks	23
4.1	Community detection workflow	23
4.2	User-oriented community detection	24
4.2.1	Probabilistic divisive community detection .	25
4.2.2	Top-sampled community search	26
4.3	Interpretable results	27

5	Experiments	28
5.1	Experimental Setup	28
5.1.1	Datasets	28
5.1.2	Algorithm implementations	29
5.1.3	Hardware setup	29
5.2	Experiment 1 (Runtime performance)	30
5.3	Experiment 2 (Iterative community detection)	31
5.4	Case Study: Detecting topic-based communities . . .	33
5.4.1	Detecting sub-topics	36
5.4.2	Community search	38
5.5	Discussion	40
6	Psychological profiling in social networks	42
6.1	Linguistic and psychographic features in social networks	42
6.2	Related research	42
6.3	LIWC	43
6.4	Inferring psychographic attributes	44
7	Conclusion	46
7.1	Future Work	47
7.1.1	Extending the node-attributes	47
7.1.2	Other data structures	47
7.1.3	Other use cases	48
	Bibliography	49

1 Introduction

Network theory is an important tool for describing and analyzing complex systems in various scientific fields. A social network is a network in the form of a graph where each node (or vertex) represents an actor and an edge between two nodes represents a relation or interaction between two actors. Social networks such as Facebook, Twitter, and Google+ has billions of users making them a gigantic source of potentially valuable data. One of the most common tasks when studying networks is that of clustering the nodes in order to detect densely linked groups that are referred to as network communities, clusters or modules. The task of clustering a network is typically referred to as *community detection*. Community detection (CD) is a well used method for analysis of social- and biological systems and the identification of network communities can discover underlying organizational principles as well as reveal nodes that share a functional property or role [1]. For example, a community in a social network may be a group of people attending the same school. Communities in biological networks emerges due to the fact that it is more likely for two proteins to interact with each other if they belong to a common functional module [2]. In this thesis, methods for community detection in large scale node-attributed networks are examined. While there have been extensive research on methods for either node-attributed networks or large scale networks, there are limited alternatives enabling analysis on large scale node-attributed networks. Most of these methods also fail to incorporate any form of high-level user preferences into the analysis.

1.1 Aim & objectives

In this thesis, two methods for user-oriented, efficient community detection in node-attributed networks are presented. The focus have been to identify communities in large scale social networks using not only the relations in a network but also considering other attributes of nodes. The common approach to community detection clusters the network based on how densely connected the nodes are, but the context of the community and the lack of interpretability gives little to no results when trying to find pre-defined types of communities. For example, an intelligence analyst working at a law enforcement agency may want to partition a network into communities where the members of a community have structural properties that motivates that they are a community, as well as they share a number of user-specified fea-

tures (such as geographic location and online activity). Traditional community detection methods either lacks the support for node-attributed networks and/or falls behind in terms of runtime performance when analyzing large networks. In this thesis, the aim is to investigate, develop and evaluate a user-oriented methodology around state-of-the-art algorithms for community detection in large scale node-attributed networks. A feasible method would enable intelligence analysts to use community detection to create communities with an underlying meaning not only based on the relations in the network. More specific, this thesis tries to answer the following questions:

- What community detection algorithm(s) are suitable for assisting analysts in their analysis of large scale node-attributed social networks?
- How can high-level user preferences be incorporated into the analysis?
- What types of communities can be detected using these algorithms?
 - Can communities that form around a specific topic be identified?

This thesis was conducted at the Swedish Defense Research Agency (FOI), a research institute conducting research in the area of defense and security¹.

1.1.1 Delimitations

This thesis focuses on a specific type of network: static large scale node-attributed networks. This means that only methods supporting node attributes and that are highly scalable are of interest. Node-attributed networks can be modeled as multi-layer networks where edges may represent attributed similarity. Methods based on the multi-layer representation are not examined. Furthermore, only methods for non-temporal networks are evaluated.

1.2 Overview of this thesis

The rest of this thesis is outlined in the following way: First, we discuss related work, detailing what they have done and how it is related to this thesis. This is followed by a theoretical background about community detection methods, which is needed to understand our contribution and results. When the theoretical background has been presented, our contribution is presented. The contribution is two new methods that are built on top of already existing community detection methods. The proposed methods are

¹<https://www.foi.se/en/about-foi.html>

applied in a number of experiments, displaying the potential and performance. From the results in the experiments, we propose how these methods could be used in a use-case where the user wants to detect communities of individuals with the same psychological profile. The use-cases for detecting these types of communities are many. This could potentially be used to detect sub-networks of people with a hostile mindset like terrorists cells, organized crime networks or child abuse networks.

1.2.1 Background

The motivation for this thesis is that traditional methods for community detection either lacks the support for node-attributed networks or suffer from bad runtime performance on large networks. Recently, methods have emerged to solve this problem. However, these methods have no built-in support for divisive clustering (iteratively finding sub-communities) or incorporating high-level user preferences into the community generation process. This makes it hard for analysts to interpret the resulting communities, and limits them to non-attributed networks or small-scale networks. Related methods such as community search methods, allows the user to post a set of exemplar nodes. This requires the user to have identified a set of exemplar nodes of similar characteristics as the desired members of the resulting communities. Even if the user identifies a set of exemplar nodes, there are no guarantees for the method to identify all probable community-members.

1.2.2 Our Contribution

We propose two methods that can be used to create communities with an underlying meaning not only based on the relations in the network. The first method allows for probabilistic divisive community detection in large node-attributed networks. This method divisively clusters a network, identifying communities and sub-communities fulfilling some constraint posted by the user. The second method allows for automatic generation of exemplar nodes for a community search algorithm. This method allows the user to specify a set of attribute associations, and will generate the nodes matching these associations as an exemplar set. These methods are tested in a number of experiments and their potential is demonstrated in an example case study.

1.2.3 Psychographic clustering of social networks

Finally, a proposal and motivation is presented, describing potential future use-cases. We believe that using the proposed methods in conjunction with a node-attributed network detailing users' psychological profile could be

of great interest of intelligence analysts at law enforcement agencies. We strongly believe that research about how these profiles can be generated for users in a social networks [3, 4, 5] could be combined with our community detection methods, in order to detect communities of people sharing psychographic traits.

2 Related work

A substantial amount of the available community detection methods do not take node attributes into consideration. Most of them either relies only on the graph structure or only on the nodes' attributes. Community detection methods only relying on graph structure, typically tries to optimize modularity, cut-ratio, or conductance. Clustering methods only relying on the attributes/features of the nodes typically tries to partition a set of feature vectors into clusters of similar feature vectors. Relying on only one of these sources of data, important information about the network might be lost. Recently, methods have been presented for detecting communities that can overlap, and thus represent the actual behavior in social networks [6, 7]. But considering that these methods ignore node attributes, they may lose interesting information about the network. Methods for overlapping community detection with node attributes have been developed in response to this. Methods such as Circles [8], CODICIL [9] and topic-models such as Block-LDA [10] are example of these kinds of methods. In 2013, Leskovec *et al.* presented CESNA [1] which aims to be a highly scalable method for detecting overlapping communities in large node-attributed networks. The authors of CESNA compared the runtime performance of CESNA and other methods for overlapping community detection, Circles, CODICIL and Block-LDA included. The authors show that CESNA has a considerable advantage in scalability compared to its competitors. CESNA also has the attractive property that: the resulting communities easily can be interpreted by analyzing the weighted attributes of the nodes belonging to each community. As discussed by Magnani *et al.* [11], one of the open problems with community detection in node-attributed networks is how to interpret the resulting communities. Other approaches to deal with large node-attributed networks are community search methods. Community search reduces the complexity by only detecting a subset of all communities in the network, given that the user has specified a set of "query-nodes". The resulting subset of communities are candidate communities of which the query nodes could be members of. Some notable algorithms for community search in large node-attributed networks are ACQ [12] and FocusCO [13]. In this thesis, we will not look into ACQ due to the lack of time.

3 Theory

3.1 Social network analysis

A social network is a representation of social structures. Social network analysis (SNA) is the process of investigating such structures, using networks and graph theory. The goal of social network analysis is to identify local and global patterns, and to examine network dynamics and influential entities. People in a social network are represented by *nodes* (also known as *actors* or *verticies*). A relation or interaction between two people is represented by an *edge* (also known as *connection* or *link*). Traditional networks G are based on a set of nodes V and a set of edges E , and notioned as $G(V, E)$. Network elements (nodes and edges) may have values associated to them such as *weights* or *attributes*. In a node-attributed network $G(V, E, F)$, each node is associated with a set of attributes (also known as *features* or *content*). The attributes are typically represented as a vector for each node, resulting in a feature matrix F for all network nodes. The attributes describes the node, not its relations, i.e. values representing age, location and/or gender. Edges in a network are typically represented as an adjacency matrix. An adjacency matrix is a square matrix representing a finite graph. The elements of the adjacency matrix indicate whether pairs of nodes are connected by an edge or not in the graph. Most networks of interest display community structure, meaning that the nodes of the network are organised into groups called *communities* (also known as *clusters* or *modules*). In social network analysis, a community can be defined as a group of nodes that can easily be grouped into a set such that they are densely connected internally. However, some types of communities are too complex to describe as "a set of densely connected nodes". Thus, communities can also be defined based on the principle that: Pairs of nodes are more likely to be connected if they are both members of the same communities. Likewise, they should be less likely to be connected if they share no common communities.

3.1.1 Community detection

Community detection (also known as: *graph/network clustering*) is a problem with no single agreed definition. Consequently, there are no clearcut guidelines on how to assess the performance of different methods or how to compare them with each other [14]. The goal of community detection methods is to group sets of nodes sharing some form of similarity, such

that some clustering metric (such as density or) is optimal. Community detection can be categorized into *overlapping* and *non-overlapping* methods. Many algorithms identify non-overlapping community structures i.e., detecting pair-wise disjoint communities. However, real life communities are not always disjoint from each other. Communities in social networks tend to overlap with each other since the active users can possibly participate in multiple groups simultaneously [15]. Community detection methods can also be categorized depending on if the community defining similarities are structural similarities, attributed similarities or a combination of both. Traditional methods grouped nodes based only on their structural similarities or only based on their attributed similarities. Most of the available methods for detecting communities based on both types of similarities are computationally expensive. Some of the traditional classes of clustering/community detection algorithms are described below.

Graph partitioning The problem of graph partitioning is to group the nodes in a graph into a predefined number k of partitions such that the number of edges going between partitions is minimal. However, it is uncommon to know the number of clusters in which the network is most naturally/efficiently split or other dictations about the membership of the nodes [16].

Hierarchical clustering Networks may have what's called a hierarchical structure. This means that the network displays multiple levels of grouping of the nodes i.e., small clusters within large clusters within even larger clusters, and so on. Hierarchical clustering starts with computing the similarities between all the nodes in the network (regardless if they are connected or not), according to some similarity measure. This results in a similarity matrix of size $n \times n$ where n is the number of nodes in the network. The aim of hierarchical clustering methods is to identify groups of nodes with high similarity, and can be classified in two categories:

1. *Agglomerative algorithms*: A bottom-up approach where clusters are iteratively merged if their similarity is high enough.
2. *Divisive algorithms*: A top-down approach where clusters are iteratively split by removing connected nodes that have low similarity

The complexity of agglomerative algorithms are in the general case $O(n^2 \cdot \log n)$ [17] which is infeasible for large scale networks. Divisive algorithms with an exhaustive search is $O(2^n)$, which is even worse.

Partitional clustering Partitional clustering is a popular class of methods. Just as for graph partitioning algorithms, the number of resulting clusters k needs to be predefined by the user. Each node in the network is seen as a point in a metric space, and a distance measure is defined between pairs of points in this space. The distance is typically a measure of dissimilarity between nodes. The goal of these algorithms is to group the nodes into k clusters, such that some given distance-based cost function is maximized/minimized. The downside with partitional clustering algorithms is that they require the user to specify the number of clusters. In addition, embedding a network in a metric space can be natural for some networks, but rather artificial for others [16].

Spectral clustering Spectral clustering algorithms treat the community detection as a graph partitioning problem. The goal of these algorithms is: Given data points $x_1, \dots, x_n$ and similarities $w(x_i, x_j)$, partition the data into groups so that points in a group are similar and points in different groups are dissimilar. Spectral clustering algorithms tend to favor connectivity over compactness. This means that the algorithm will cluster the nodes that are connected, but not necessarily compact within convex boundaries. Spectral clustering algorithms make use of the eigenvalues of the similarity matrix of the data, in order to reduce the dimensionality and perform the clustering in fewer dimensions. The downside of spectral clustering is that they are sensitive to the choice of similarity function w and parameters. Spectral clustering algorithms are also computationally expensive for large datasets.

3.2 Community detection with node-attributed graphs

3.2.1 Node-attributed networks

In order to identify network communities, most methods will try to cluster the set of nodes in the network, so that each cluster contains nodes of similar characteristics. In most real-life networks there are two sources of information that can be used to perform the clustering task. The first is information about the nodes, often referred to as *node attributes*. The attributes may typically describe features of a single actor in a network. In a social network these attributes may be: age, geographical location or gender. The second source is the set of connections/edges between the nodes in the network. This describes the relations or interactions between the actors in a network. In a social network, a connection between two nodes could represent friendship, following or directed communication between two people. Figure 3.1 illustrates a regular network and a node-attributed network.

Figure 3.1: A regular network and a node-attributed network (both are undirected).

Traditional clustering methods typically focus on only one of these sources of information. Clustering algorithms focusing on attribute data, group the nodes in the network into sets of nodes whose attributes are similar [18, 19]. These algorithms ignore the relationships (edges) in the network. In contrast, traditional community detection algorithms find communities based on the connectivity structure of the network [20, 21]. This way, nodes will be grouped depending on how densely connected they are with other nodes or other structural properties. These algorithms tend to ignore the node-attribute data. Algorithms ignoring one of these two sources of information can result in a loss of important information about the network. For example, if node attributes are ignored, it may be hard to tell which community a node with few connections belongs to. Looking at just the network structure will most likely not give us enough information to determine community membership. Likewise, the network structure might indicate that two nodes belong together even though they do not share any attributed similarities or if one of them is missing attribute information. Combining both sources will result in communities who are densely connected, and whose nodes share some common attribute(s) [1]. Combining information about the attributes with the network structure leads to a more precise detection of network communities. Each source of information can also complement the other source in case of noisy or missing data. Node attributes can also explain the detected communities in terms of what factors the community emerged from. In Figure 3.2 a matrix representation of the edges (called an adjacency matrix) in a Facebook network is shown. There is a column and a row for each node in the network and a positive value indicates that an edge exists between those nodes. In the adjacency matrices in Figure 3.2, you can see a ego-network (a network surrounding a single node) from Facebook. The right-most is plotted without any sorting of the rows/columns while the other two are sorted after attributed similarity. The pink-colored area indicates not having any attribute-information while the

teal-colored areas indicates that the people share one or more attribute similarity (depending on which area). The dot-density within an area indicates the number of edges between the people. Looking at the density of the areas where the nodes have attributed similarities, we can see that they tend to be more densely connected. People often connect with each other due to some shared attribute between them. For example, people who are attending the same school or works at the same office, are likely to be connected with each other.

Figure 3.2: Adjacency matrix for a ego-network from Facebook. The right-most is plotted without any sorting while the other two are sorted after attributed similarity.

Recently, methods for detecting communities based on both structural and node attributes have started to emerge. Some of these methods use single-assignment clustering [22, 23, 24, 25]. Single-assignment clustering does not allow a node to belong to more than one community, making overlapping communities non-detectable. Other methods are based on topic models which supports detection of overlapping communities [26, 27, 28, 29]. These methods assume "soft" node community memberships. This means that while they do allow for nodes to be part of multiple communities, nodes may not simultaneously have high "membership-strength" to multiple communities [1]. This may not be a appropriate assumption since a node may have high membership strengths to multiple ground-truth communities. For example, a person should be able to have high membership strength to both its classmates and its childhood friends. This makes the number of available methods and algorithms limited, especially if the method should be able to handle large networks.

3.3 Community detection in large scale networks

Community detection is used with the hopes of getting an understanding of the hierarchical organization of the network, by identifying modules (called communities), using the information encoded in the network. These communities can give a better understanding of the network by analyzing the

properties of the modules and the nodes belonging to it. Community detection methods have been thoroughly studied for small networks [30, 31]. However, with the fast development of online social networks that have many millions of active users, (e.g Twitter with more than 300 Million monthly active users¹) there is an increasing need for algorithms that can handle large networks. Identifying interesting communities in large scale networks is infeasible with most of the well-established community detection methods. Some methods have been developed for large scale networks [2, 32], but few have support for node-attribute data. One of the biggest challenges with large node-attributed networks is that for each attribute introduced in the network, the number of attribute value combinations increases exponentially. This creates computational challenges for the clustering process [11]. In order to solve the computational complex challenges, algorithms such as CESNA [1] employs a probabilistic approach. This allows for a computationally less complex clustering of the network. How CESNA works is later described in this report. Another approach to reduce the complexity of detecting communities in large node-attributed networks is community search (CS). Community search reduces the complexity by only detecting a subset of all communities in the network, given that the user has specified a set of "query-nodes". Some notable algorithms for community search in large node-attributed networks are ACQ [12] and FocusCO [13].

3.4 Community detection vs. community search

In social network analysis, community detection (CD) is the problem of identifying communities, trying to maximize some form of metric. With the increase of studies related to big data, a similar problem called community search (CS) has been defined. Community search can be seen as a variant of the community detection problem. The major difference is that community search is query-dependent. The goal of community search is to: given a query node q , find the most likely community of which q is a member. The difference between CD and CS is that CD finds communities in the entire network while CS finds one or more communities that a queried node is most probable to be a member of [33]. In some cases though, you may not have access to one or multiple query nodes matching the desired cluster(s). In some use cases, explicitly stating which attributes the resulting communities should be associated with, can be more feasible in comparison to providing exemplar nodes. For example, the user may want to partition the entire network and find all communities/sub-communities with some characteristics. CS methods will only find communities related to the exemplar nodes,

¹<https://about.twitter.com/company>

which may not capture all communities with the desired characteristics. In order for CS methods to find all these communities, it requires exemplar nodes from each community.

Not all traditional classes of community detection algorithms has support for node-attributed networks. Algorithms for node-attributed network tend to be computationally expensive for large networks. In the following three sections, we describe three algorithms supporting large node-attributed networks.

3.5 CODICIL

Ruan *et al.* [9] presented a method for efficient community detection in large networks with content/attribute-information. The developed method, CODICIL (**C**ommunity **D**iscovery Inferred from **C**ontent **I**nformation and **L**ink-structure), is a method combining node attributes and edges in graph structures for the purpose of community discovery. CODICIL uses a "biased edge sampling procedure" which generates a new set of edges between nodes with attributed similarity. Adding the newly generated "content edges" to the original network, CODICIL can cluster the network by taking both node-attributes and network structure into consideration. CODICIL's approach is the principle of "fusing" edge structure and node attributes by creating an additional set of edges between nodes with attributed similarities. The fusing is a transformation of the original node-attributed network to a non-attributed network (a regular graph) with a new set of edges. This sampled graph can subsequently be clustered with any graph clustering algorithm [9]. The rest of this section contains a summary of the workings of CODICIL.

Algorithm overview Let the original network G_t be an undirected node-attributed graph $G_t = (V, \varepsilon_t, \tau)$ with n nodes $v_1, \dots, v_n \in V$, and a set of edges connecting the nodes ε_t and a collection of n corresponding term vectors $t_1, \dots, t_n \in \tau$. A term vector corresponds to the attributes describing a node.

Let G_{sample} be a simplified, edge-sampled graph of G_t : $G_{sample} = (V, \varepsilon_{sample})$

G_{sample} has the following properties:

- G_{sample} has the same node set as G_t . This means that no nodes are added or removed when simplifying the graph.

- $|\varepsilon_{sample}| \ll |\varepsilon_t|$, as this enables both better runtime performance and lower memory usage in the subsequent clustering stage
- The resulting edge set ε_{sample} will connect node pairs that are similar both structure-wise and attribute-wise. This means that edges may be added, that are not in original edge set ε_t .

Given a network’s adjacency matrix and attributes for the network’s nodes, the CODICIL algorithm has three main steps:

1. Creating new edges based on attributed similarities between nodes, called ”content edges”.
2. Sample the union of content edges and the original topological edges with bias, retaining only edges that are relevant in local neighborhoods.
3. Partition the simplified graph into clusters.

The CODICIL framework takes care of the two first steps, leaving the actual clustering of the simplified, edge-sampled graph to the user.

Creating content edges The process for creating ”content edges” can be described in the following way: For each node v_i , its k most attribute-similar neighbors are computed. This process is referred to as the *TopK* sub-routine. For each of these top- k neighbors v_j , an edge connecting v_i and v_j is added to content-edge set ε_c . To calculate the attribute-similarity of nodes, CODICIL calculates the cosine similarity of t_i ’s TF-IDF (term frequency-inverse document frequency) vector. Other similarity measures could be used, but cosine similarity is proposed in the original paper. TF-IDF is a statistic intended to reflect the importance of a word to a document in a collection or a corpus. CODICIL computes each term-vector’s TF-IDF vector. For a content/attribute unit c , its TF-IDF value in a term vector t_i is computed in the following way:

$$tf-idf(c, t_i) = \sqrt{tf(c, t_i)} \cdot \log \left(1 + \frac{|T|}{\sum_{j=1}^{|T|} tf(c, t_j)} \right)$$

The cosine similarity of two vectors v and u is:

$$cosine(v, u) = \frac{v \cdot u}{||v||_2 \cdot ||u||_2}$$

The most similar neighbors of v_i are the k nodes with the highest TF-IDF vector cosine similarity values (in comparison with v_i).

Local ranking of edges and graph simplification Given the edge union ε_u of the original edges ε_t and the newly created content-edges ε_c , ε_{sample} is created by retaining the most relevant edges from ε_u . For each node v_i , the edges to retain are selected from its local neighborhood in ε_u . CODICIL then computes the topological similarity between node v_i and its neighbor v_j as the overlap of their respective topological sets $I = ngr(v_i, \varepsilon_t)$ and $J = ngr(v_j, \varepsilon_t)$, using either cosine similarity or Jaccard coefficient. The Jaccard coefficient is defined as:

$$jaccard(I, J) = \frac{|I \cap J|}{|I \cup J|}$$

After the computation of this topological similarity vector sim_i^t , it is normalized so that it either scales the vector to $[0,1]$ (zero-one), or using the z-norm which centers and normalizes values to zero mean and unit variance.

$$z\text{-norm}(\vec{x}) = \frac{x_i - \hat{\mu}}{\hat{\sigma}}, \quad \hat{\mu} = \frac{\sum_{i=1}^{|\vec{x}|} x_i}{|\vec{x}|}, \quad \hat{\sigma}^2 = \frac{1}{|\vec{x}| - 1} \sum_{i=1}^{|\vec{x}|} (x_i - \hat{\mu})^2$$

CODICIL compute v_i 's content/attribute similarity by calculating the similarity on the term vectors t_i and t_{v_j} . The topological and content similarities of each edge is then aggregated with a specified weight α . Finally CODICIL retains the edges with the highest similarity values. While CODICIL does not do any clustering, it generates a new graph where new edges are created for attribute similarity. This makes CODICIL flexible in what types of communities it can find, since you can use any graph clustering algorithm on the generated graph. The drawback of CODICIL is its computational complexity. The overall runtime complexity of CODICIL can be expressed in the following way:

$$O(P + n^2 \cdot \log n + C)$$

Where n is the number of nodes in the network, P is the edge pre-processing time (complexity dependent on the implementation on the TopK operation) and C is the complexity of the clustering algorithm used on the generated graph. The authors of CODICIL displayed in the original paper [9] that when clustering a CODICIL sampled graph, they achieved a higher average F-score compared to methods such as LDA, PCL-DC, and K-means. In the original paper, Multi-Level Regularized Markov Clustering (MLR-MCL) [34] was used to cluster the generated graph. In this paper, regular Markov Clustering (MCL) [35] is used.

3.6 CESNA

Communities from Edge Structure and Node Attributes (CESNA), is an algorithm developed with the aim to accurately and efficiently detect communities in node-attributed networks [1]. CESNA has a probabilistic approach to community detection and supports binary-valued attributes. CESNA is developed under the assumption that communities "generate" both the network structure and the content (in the form of node attributes). Some of the most attractive properties of CESNA are:

1. CESNA detects overlapping communities.
2. CESNA assigns hard node-community memberships. More specifically, CESNA allows a node to have strong membership to multiple communities. This means that the sum of membership strengths for all communities c for a node u does not need to be a constant value:
$$\sum_c F_{uc} \in [0, \infty)$$
3. CESNA allows for dependence between the edge structure and the node attributes. This means that both the node attributes and the edge structure affects the resulting communities.

Inferring network edges CESNA employs the same probabilistic generative process for generating an adjacency matrix as its predecessor BigCLAM [32]. BigCLAM is an overlapping community detection method for large networks. One of the core parts of BigCLAM is its Cluster Affiliation Model. This model is built on three main ingredients:

1. Communities arise from shared group affiliation. BigCLAM represents node community memberships using a bipartite affiliation network model that links nodes of the network to communities that they belong to. In this affiliation network model, the nodes of the network are affiliated with the communities they belong to. Edges for the underlying social network are then derived based on the nodes' community affiliations.
2. People are involved in communities to various degrees. BigCLAM assumes that each affiliation edge in the bipartite affiliation network has a non-negative *weight*. Where a higher affiliation weight between a node and a community indicates a higher likelihood of the node being connected to other members of that community.

3. When people share multiple community affiliations, the links between them stem from one dominant reason. This means that for each community a pair of nodes shares, we get an independent chance of connecting the nodes. This results in that the more communities two nodes share, the more likely they are to be connected.

Given the bipartite community affiliation $B(V, C, M)$, BigCLAM can generate a sample network $G(V, E)$. To do this, BigCLAM considers a simple parameterization where it assigns a nonnegative weight F_{uc} between node $u \in V$ and community $c \in C$, where $F_{uc} = 0$ means that node u has no affiliation to community c . Given F , BigCLAM assumes that each community c connects its member nodes depending on the value of F_{nc} for all member nodes n . Where the probability of a community c connecting nodes u and v is: $1 - \exp(-F_{uc} \cdot F_{vc})$. Since each community c connects the nodes u and v independently the edge probability of (u, v) is: $1 - \exp(-\sum_c F_{uc} \cdot F_{vc})$. BigCLAM does not use a gaussian distribution or logistic link function, instead it tries to optimize the model likelihood of explaining the edges in the observed network. In summary: Given a non-negative matrix F where F_{uc} is a weight between node $u \in V$ and community $c \in C$: BigCLAM generates a sample graph $G(V, E)$ by creating the edge (u, v) between a pair of nodes $u, v \in V$ with the probability: $p(u, v) = 1 - \exp(-F_u \cdot F_v^T)$, where F_u is a weight vector for node u .

Inferring node attributes Just as CESNA uses the probabilistic generative process of BigCLAM to model network edges from community affiliations, node attribute values can be modeled as well. CESNA assumes the attribute to be binary-valued, and for each attribute X_{uk} , a separate logistic model is considered. CESNA works with the intuition that, given a node's community memberships, it should be possible to predict the value of each of the node's attribute values. Thus, CESNA regards community memberships $F_{u1}, \dots, F_{uC}$ as input features of the logistic model with the associated logistic weight factor W_{kc} for each attribute k and each community c . The logistic weight factor W_{kc} model parameter for community c and the k -th attribute. W_{kc} is real-valued and represents the relevance of each community membership c to the presence of the attribute k . In Figure 3.3 you can see a plate representation of CESNA, made by the original authors [1].

Figure 3.3: Plate representation of CESNA. X_{uk} : k -th attribute of node u ; W_k : Logistic weight vector for attribute k ; Q_{uk} : Probability that $X_{uk} = 1$; F_{uc} : Membership strength of node u to community c ; A_{uv} : Indicator for whether the nodes u and v are connected; P_{uv} : Probability that $A_{uv} = 1$.

To detect communities, given a graph G and node attributes X , CESNA will try to find the most likely community affiliation factor matrix $\hat{F}$ and logistic weight factor matrix $\hat{W}$.

$$\hat{F}, \hat{W} = \operatorname{argmax}_{F \geq 0, W} \log P(G, X | F, W)$$

In order to prevent this function from overfitting, regularization is introduced. Regularization is generally used as a technique to improve the generalizability and solve the overfitting problem in statistical models. The idea is that making assumptions resulting in overfitting are unlikely to generalize well to future data. These assumptions may happen to explain a few data points from the current data well, but this may be because of accidents of the sample. Introducing regularization, we get:

$$\hat{F}, \hat{W} = \operatorname{argmax}_{F \geq 0, W} \mathcal{L}_G + \mathcal{L}_X - \lambda |W|_1$$

Where:

$$\begin{aligned}\mathcal{L}_X &= \sum_{u,k} X_{uk} \cdot \log(Q_{uk}) + (1 - X_{uk}) \cdot \log(1 - Q_{uk}) \\ \mathcal{L}_G &= \sum_{(u,v) \in E} \log(1 - \exp(-F_u F_v^T)) - \sum_{(u,v) \notin E} F_u F_v^T \\ Q_{uk} &= \frac{1}{1 + \exp(-\sum_c W_{kc} \cdot F_{uc})}\end{aligned}$$

and λ is a regularization hyperparameter. l1-regularization is invoked on W to avoid overfitting and to learn sparse relationships between communities and attributes.

The optimization procedure is further described in the original paper [1].

A full iteration of CESNA takes $O(|E| + NK)$ operations, which is linear in the number of edges E , nodes N and the number of attributes K . Introducing a new node to the network does not necessarily mean that you need to introduce a new type of attribute to the network. In practice, the complexity in relation to the number of nodes n is $O(n)$, as long as you have a fixed attribute space for the network. This is a very reasonable complexity for large scale analysis, making CESNA a good choice for large networks. The authors also displayed in the original paper that CESNA yielded a higher F_1 -score compared to methods such as Circles, CODICIL, Block-LDA, and BigCLAM [1]. Comparing CESNA to community detection methods that takes both network structure and node attributes into consideration (CODICIL, Block-LDA, and Circles), the authors observed an average of 47% relative improvement in the accuracy of detected communities.

3.7 FocusCO

In 2014, Perozzi *et al.* published a paper called "Focused Clustering and Outlier Detection in Large Attributed Graphs" [13]. The authors presented an algorithm for community search and outlier detection called FocusCO. FocusCO is a user-oriented approach for mining large attributed graphs.

In this thesis, we do not utilize the outlier detection of FocusCO. Thus, the theory of the outlier detection will not be explained. The type of community detection problem FocusCO tries to solve can be defined in the following way: Given an attributed graph $G(V, E, F)$ with $|V| = n$ nodes, $|E| = m$ edges, where each node is associated with $|F| = d$ attributes. Extract only the type of clusters from G pertaining the user u 's interest, rather than partitioning the whole graph. The user specifies the interest by supplying

a set of exemplar nodes C_{ex} that the user considers to be similar to the type of nodes the clusters of interest should contain. FocusCO makes the assumption that the cluster emerges around a few defining attributes of its members. Using this assumption, FocusCO aims to infer the implicit weights β_u (relevance) of the attributes that define the nodes in the set of exemplar nodes C_{ex} , i.e. the weights of the attributes that makes them as similar as possible. It is expected for β_u to be a sparse vector with large weight for only a few attributes. These few attributes with large weights are referred to as *focus attributes*. The process of how FocusCO does this is described later in this section. After inferring β_u , FocusCO tries to extract focused clusters C from G that are both structurally dense and separated from the rest of the network, as well as consistent on the focus attributes. The extracted clusters can be overlapping, meaning that nodes may be members of multiple clusters. The set of resulting clusters C is a subset of all the clusters in G since different sets of clusters are expected to emerge from different attributes. Because the aim is to extract the clusters who are similar to the exemplar nodes, we will not extract clusters not similar to the exemplar nodes.

3.7.1 Algorithm details

Inferring attribute weights the first step is to identify the relevance weights of the attributes that the exemplar nodes have in common.

Our first goal then is to identify the relevance weights of node attributes that make the exemplar nodes similar to each other. Computing this similarity is often captured by the (inverse) Mahalanobis distance: The distance between two nodes with feature vectors f_i and f_j is: $(f_i - f_j)^T A (f_i - f_j)$. Setting A as the *identity matrix* yields Euclidean distance, otherwise the features/dimensions are weighted accordingly. Now we have a distance metric learning problem: Given the exemplar nodes, how can we learn A such that they end up having small distance to each other? FocusCO adopts the following optimization objective:

$$\min_A \sum_{(i,j) \in P_S} (f_i - f_j)^T A (f_i - f_j) - \gamma \log \sum_{(i,j) \in P_D} \sqrt{(f_i - f_j)^T A (f_i - f_j)}$$

This is convex and enables local-minima-free algorithms to solve it, especially for a diagonal solution. P_S and P_D are two sets of similar and dissimilar pairs of nodes. P_S is constituted by all pairs of exemplar nodes. P_D is created by randomly drawing pairs of nodes that do not belong to the exemplar set. When constructing P_D , samples may be drawn that are similar to those in P_S . In order to alleviate the effect of these types of draws, the size of P_D needs to be kept sufficiently large. This is true, assuming that the number of

dissimilar pairs are substantially larger than the number of similar pairs in the original distribution. When inferring attribute weights, FocusCO learns a diagonal A matrix. This choice have been made for two reasons. First, individual weights for attributes provide ease of interpretation. Second, learning a diagonal A is computationally more feasible (especially in high dimensions), compared to learning a full one. This is because learning a full matrix requires solving a program with semi-definite constraint.

Extracting focused clusters Once the attribute weights β have been determined, FocusCO extracts the focused clusters of interest. The main idea behind this extraction of focused clusters is to first identify good candidate nodes that potentially belong to such clusters. This is followed by an expansion round these candidate nodes, in order to find clusters. To find the candidate nodes, the graph's edges are reweighed using the focus attributes:

for each $(i, j) \in E$ **do:**

$$w(i, j) = \frac{1}{1 + \sqrt{(f_i - f_j)^T \cdot \text{diag}(\beta)(f_i - f_j)}}$$

end

After the edges has been reweighed, only the edges with a high weight are kept. Components of nodes being connected by high-weighted edges are then used as seed sets for the expansion. The next step is to expand around the kept sets. In this step the clustering objective is conductance $\phi^{(w)}$ weighted by focus. This is done by computing the fraction of the sum over all edges with one vertex in C and one vertex outside of C , divided by the sum over all nodes in C and the sum of all edges from that node. The formula for the focus-weighted conductance is:

$$\phi^{(w)}(C, G) = \frac{W_{\text{cut}}(C)}{W_{\text{Vol}}(C)} = \frac{\sum_{(i,j) \in E, i \in C, j \in V \setminus C} w(i, j)}{\sum_{i \in C} \sum_{j, (i,j) \in E} w(i, j)}$$

where $W_{\text{Vol}}(C)$ is the total weighted degree of nodes in C . The lower conductance a cluster has, the higher quality of few cross-cut edges and high density within the cluster. The expansion procedure from a seed set starts with enlisting all their non-member neighbors as the candidate set. For each candidate node n , the difference $\Delta\phi_n^{(w)}$ in cluster conductance is computed, to see if n is to be added to C . If there exist any node with a negative Δ which improves node conductance, the best n is picked, with the minimum. This results the the largest absolute drop in conductance. This keeps on iterating until no candidate node yields a negative Δ . This

node-addition strategy is based on a best-improving search strategy. While this process is cautious about which node to add (only the best one at every step), the decisions are still greedy. This process keeps on repeating until convergence (when conductance stops changing). This process is guaranteed to converge as the weighted conductance of a cluster is lower-bounded by 0 and the weighted conductance is decreased in every iteration. The operation of adding or removing a node u from a cluster S has complexity proportional to the degree of u i.e. $O(d(u))$. Additionally, the total volume of S is increase/decreased by the weighted degree $w(u)$ of u , when it is added or removed, taking $O(1)$ [13]. The authors displayed in their original paper [13] that the precision of FocusCO is often higher than its recall. The authors also displayed that FocusCO outperformed a method called CoDA. The authors conclude that their experimental results show the robustness of FocusCO, where its performance remains quite stable across different settings.

3.8 Ethical aspects

The users in a social network typically corresponds to real people or organizations. Thus, analyzing a social network includes analysis of individuals. This brings legal and ethical aspects to social network analysis. Social networks may contain vast quantities of naturally-occurring data on many different topics for example, consumer behavior, political views and personal preferences. However, it is important to keep in mind that the data may contain sensitive personal information and therefore it is important to ensure that the data is obtained according to the highest ethical standards [36]. The many ethical issues can be summarized into three categories.

1. Fair data One aspect regards the use of fair data. Many organizations use publicly available data for their analysis. While this may seem fair, the definition of publicly available data is ambiguous. Many social network services, such as Twitter and Facebook, have public API's for retrieving data about their users. The policy for what data the social network service will share publicly, differs from service to service. Most service either adopts a opt-in or opt-out policy for sharing data publicly. These policies lets each user either share data by default and actively opt-out if they disagree *or* not sharing data publicly by default and actively opt-in if they want to share data publicly. When registering an account at a social network service, the user must typically agree to the service's policies. Many users agree to the terms and policies set by the social network service without thoroughly reading them. This leads to many users sharing data publicly without knowing it.

Some social network services may not share (some or any) public data to third parties. This leads to third parties using unapproved methods (such as web-crawling) for retrieving data about the service’s users. This is a violation both to the social network service’s terms of use for their service, as well as the user’s consent to share data. Some social network services restricts the intended use for the publicly shared data. Twitter recently updated the privacy policy for their public API² stating: *”We prohibit developers using the Public APIs and Gnip data products from allowing law enforcement - or any other entity - to use Twitter data for surveillance purposes.”*

2. Fair usage There are numerous use cases for social network analysis. Some examples are: targeted marketing, surveillance, sociological studies and recommendation engines. Some use cases raises more concerns than others. Many users may be glad to have movie recommendations that suit their personal profile, while feeling uneasy with law enforcement doing automated surveillance. Therefore, researchers and other who study social networks should ideally have a formal consent from participants in a study. This is however not feasible in practice because of how time-consuming the process would be, and how counter-intuitive it is to the idea of big data analysis. Another solution is to anonymize the data, for example: by applying a hash function on user names, addresses, locations etc. This solution works as long as the study that is conducted consider such features irrelevant.

3. Fair conclusions The last category is about the conclusions made from the gathered data. For example, The result from a study may indicate that people with feature X are more likely to be well-educated. This may be problematic if the gathered data is biased or the methodology of the study may be flawed. Concluding that people with the absence of feature X thus are not educated is controversial. Feature X may be absent in one persons online communication but present in real life.

The categories discussed above does not cover all the ethical aspects. In this thesis, the *Framework for ethical research with social media data* [36] was used as a guideline for the work. In [36] Townsend and Wallace describes a framework for ethical use of social media data. The guide highlights ethical aspects which are taken into consideration for this thesis, but not further discussed.

²<https://blog.twitter.com/2016/developer-policies-to-protect-people-s-voices-on-twitter>

4 Probabilistic divisive clustering of node-attributed networks

In this chapter, two novel methods for user-oriented community detection in node-attributed networks are presented. The methods are designed for large-scale networks and based on principles from divisive hierarchical clustering algorithms as well as community search algorithms.

4.1 Community detection workflow

To perform community detection analysis on node-attributed networks, a general workflow is defined in Figure 4.1. The workflow consists of four steps:

Figure 4.1: General workflow

1. *Data Gathering*. This is the first step and involves the gathering or retrieval of data to analyze. The data may be an already existing dataset or it may need to be gathered from a real social network. The goal of this step is to gather semi-structured data of a social network.
2. *Feature Extraction*. The second step is to extract features or attributes for the actors in the gathered data. In the most simplest case, a set of attributes would be selected from the user's profile. If the user is performing an analysis of topic-based communities, this step would involve assigning topics to each user, based on his/hers interests.
3. *Format Conversion*. Once the data has been gathered and features has been selected, the data needs to be converted to an appropriate

format, suiting the selected clustering method. Most scalable community detection methods for node-attributed networks requires the features/attributes to be binary-valued. This requires the user to discretize the selected non-binary-valued attributes into ranges. In this thesis, we have manually set the resulting ranges for the attribute discretization. An example of this conversion can be seen in Figure 4.2. The discretization may have a big impact on the end-results. Problems like overfitting and underfitting may arise if the ranges are constructed too narrow or too broad.

4. *Analysis.* Once the data has been converted to the appropriate format, a community detection algorithm can be applied to the data. The resulting communities can then be studied and evaluated. In this step, the user can experiment with algorithm parameters and compare the different outcomes.

<i>UserID</i>	<i>Age</i>	<i>Location</i>
1	28	Sweden
2	43	Germany

<i>UserID</i>	<i>Age 0-20</i>	<i>Age 21-40</i>	<i>Age 41-60</i>	<i>Age 61+</i>	<i>Germany</i>	<i>Sweden</i>
1	0	1	0	0	0	1
2	0	0	1	0	1	0

Figure 4.2: Attribute Discretization. The upper table displays the non-discretized data. The lower table displays a possible discretization of the data, where the attribute *age* has been divided into ranges and *location* has been categorized. The value 1 indicates the possession of an attribute.

4.2 User-oriented community detection

Both CESNA and CODICIL are classified as community detection algorithms but neither CESNA nor CODICIL have any built-in functionality for letting the user steer the analysis in a query-like fashion. For example, if a user wants to detect communities where the member nodes have a specified combination of attributes, or sub-communities within a community where certain attributes are highly associated. This problem can be defined in the following way: given that the member nodes of community c has attribute a_1 , find sub-communities C' within community c that also has attribute

a_2 . Alternatively, the algorithm could automatically find the most probable communities within c , not posting any constraints on the presence of a_2 .

To solve this problem, we present two methods. The first method we call *probabilistic divisive community detection*, which is powered by the probabilistic approach of CESNA. This method is developed for large node attributed networks, letting the user incorporate his/hers high-level preferences and yields easily interpretable results. The second method we call *top-sampled community search*, which is a method for automatically generating a good pair of exemplar nodes given a query Q of desired attribute associations. This method is powered by FocusCO.

4.2.1 Probabilistic divisive community detection

Divisive clustering algorithms uses a top-down approach where clusters are iteratively split by removing connected nodes that are not similar enough. While these algorithms are well designed for finding sub-communities, they tend to be computationally expensive. Our method utilizes CESNA’s probabilistic approach for the community detection and only splits clusters with an association, specified by the user. Our method will automatically find sub-communities within the first community that is associated with the combination of attributes, specified by the user. The user’s query Q is a set with attributes and an indication whether a community should have an association to that attribute or a negative association to that attribute.

- Let G be a node-attributed network $G(N, E, A)$ where N is the set of nodes in the network, E is the set of edges in the network, represented by pairs of nodes $(i, j) \in E$ and A is the set of node-attribute pairs (n, a) for all nodes $n \in N$ and each of their associated attribute a .
- Let c be a set of nodes representing a single community and C be the full set of communities $c \in C$ yielded by one iteration of CESNA on a network G .
- Let Q be the user’s query of attributed association. Each attribute-association specified $q \in Q$ is either positive or negative $q \in \{-1, 1\}$.
- Let W_{ac} be the association weight of an attribute a to a community $c \in C$. New weights are generated for each iteration of CESNA.
- Let the similarity symbol $\sim$ represent the agreement of association weight W_{qc} and the queried association $q \in Q$, such that attribute weight W_{qc} has a significant positive/negative weight that matches the queried association for attribute $q \in Q$.

In order to iteratively detect communities within a community matching the specified query, a new network $G'(N', E', A')$ is generated from the member-nodes of the matching communities C' :

$$C' = \{c \in C \mid W_{qc} \sim q, \forall q \in Q\}$$

$$N' = \{n \in c, \forall c \in C'\}$$

$$E' = \{(i, j) \in E \mid i \in N' \wedge j \in N'\}$$

$$A' = \{(n, a) \in A \mid n \in N'\}$$

return $G'(N', E', A')$

The new network G' can then be further clustered, using a new query Q' . The ordering of multiple queries will affect the resulting clusters, since nodes are removed from the network for each iteration. The user also has the option omit the query, letting CESNA find the most probable sub-communities without any constraints.

4.2.2 Top-sampled community search

Another way to incorporate high-level user-preferences into the analysis is to use community search methods instead of community detection methods. However, this requires the user to provide at least one example node matching the desired communities. This does not allow for prioritization of the attributes related to the sample node(s) nor stating which attributes that are of interest. However, this may be useful in cases where a well defined sample has been retrieved and the user wants to analyze the surroundings of the sample. In this thesis, we present a method for generating a set of exemplar nodes, matching a query Q of attribute associations. The set of exemplar nodes is then used as the input for the community search method FocusCO. The method to generate a set of exemplar nodes is described below:

- Let Q be a user-query of attribute associations, represented as a vector of attribute associations $q \in Q$.
- Let k be the number of nodes to generate for the exemplar set.
- Let N be the set of nodes in the network, where each node $n \in N$ is represented as a feature vector corresponding to its attribute values.
- Let $TopK(V, v_q)$ denote the procedure of selecting the k vectors from a set V of vectors, that are the most similar to the vector v_q (using cosine similarity as a similarity-measure).

- Let $FeatureSelection(v, v_q)$ denote the procedure of reducing the number of dimensions/features on vector v , by only selecting the features that are present in v_q .

```

 $N_f = \{FeatureSelection(n, Q), \forall n \in N\}$ 
 $N_{ex} = \{n \in N \mid n \in TopK(N_f, Q)\}$ 
return  $N_{ex}$ 

```

This generates an exemplar set N_{ex} of k nodes with high similarity to the user-queried attribute association Q . This is done by generating a new set of vectors N_f where attributes not present in the query Q are omitted from the similarity measurement in $TopK$. The resulting set of exemplar nodes N_{ex} is then used as input to the FocusCO method. FocusCO will identify multiple set of nodes similar to the exemplar nodes, both in terms of attributed similarity, but also based on the network structure. Given the node-attributed network $G(N, E, A)$ and the set of exemplar nodes $N_{ex} \subset N$, FocusCO will return a set of communities C of which the nodes in N_{ex} are the most likely to be members of.

4.3 Interpretable results

One big challenge with community detection in general, is the interpretability of the resulting clusters. Analyzing the nature of why a community exists is a difficult problem. When CESNA produces the resulting communities, the weights for attributes associated to each community is generated as well. Studying these weights, makes it much easier to see what the community represents.

5 Experiments

In order to evaluate the methods described in the previous chapter, they were applied in a number of experiments. The methods and algorithms are hard to evaluate in terms of typical cluster evaluation metrics such as accuracy or harmonic mean of precision and recall (F_1 -score). This is partly because of the lack of ground truth communities in our dataset. It is also difficult to compare the resulting communities to "known communities" because of the nature of communities. Communities can be assembled based on different characteristics where no assemble is a more true community than the other. Some characteristics are more "natural" for social communities, and the aim is to find community detection methods that clusters that network on such natural characteristics. The aim of this thesis was to examine the characteristics our methods succeeded in detecting. In particular, can communities formed around topics of interest be detected?

5.1 Experimental Setup

Our setup consists of commodity hardware and three datasets summarized in Table 5.1

5.1.1 Datasets

The three datasets we use for our experiments are all based on the Twitter network surrounding the account **@SAPOsverige**. However, they all have different characteristics in terms of different types of node-attributes.

SAPO Twitter The SAPO Twitter dataset is the Twitter network surrounding the account **@SAPOsverige**, belonging to the Swedish Security Service (SPO). The dataset contains 80% (or 42.000) of SAPOsverige’s followers, as well as all the other accounts these 42.000 are following. The data is publicly available, but usernames and all identity revealing attributes have been anonymized with respect to the users’ privacy. The dataset contains edges corresponding to following and node-attributes such as location, account creation date, and number of tweets.

SAPO Hashtags The SAPO Hashtags dataset is a subset of the SAPO Twitter dataset, containing a subset of the original nodes. The 250 most frequently used hashtags were identified for this network. These hashtags

are used as attributes for the nodes in the network. A user is assigned a hashtag attribute if they have used it in their latest 200 tweets.

SAPO LIWC The SAPO LIWC dataset is another subset of the SAPO Twitter data. This dataset contains 285 users. For each user in the dataset, there are 10 attributes related to that user’s psychological profile, each having 5 levels. This results in a attribute space of 50.

<i>Dataset</i>	<i>Nodes</i>	<i>Edges</i>	<i>Attribute entries</i>	<i>Attribute space</i>
SAPO	1.728.501	9.403.144	6.969.266	27
SAPO-Hashtags	45.752	547.054	388.303	250
SAPO-LIWC	285	1142	2.850	50

Table 5.1: Dataset descriptions

Table 5.1 lists the datasets used in our experiments. In the table *Dataset* denotes the name of the network, *Nodes* the number of nodes in the network, *Edges* the number of edges in the network, *Attribute entries* the number of occurrences of all different attributes, and *Attribute space* the number of different attributes in the dataset.

5.1.2 Algorithm implementations

In the conducted experiments, the algorithms CESNA, CODICIL, and FocusCO was compared. The implementation for each algorithm is written by the corresponding author(s). This comparison is perhaps not fair since some of the implementations are more well-polished than others. CESNA is available via the Stanford Network Analysis Platform (SNAP)¹. It is the most polished implementation, being written in C++ and utilizing multiple threads. The CODICIL implementation² is a combination of Python and C. The FocusCO implementation³ is written in Matlab and Java and is said to be more useful as an example than a finished product.

5.1.3 Hardware setup

All experiments were performed on a single machine with commodity hardware. A specification for the machine used in the experiments can be seen

¹<http://snap.stanford.edu/>

²<https://sites.google.com/site/stochasticflowclustering/graphpreprocessing>

³<https://github.com/phanein/focused-clustering>

in Table 5.2.

CPU	Intel Core i7-6820HQ CPU @ 2.70GHz
CPU Architecture	x86_64
Cores	4
Threads per Core	2
System memory (RAM)	16 GB
L1 cache size	128 KB
L2 cache size	1 MB
L3 cache size	8 MB

Table 5.2: Machine Specification

5.2 Experiment 1 (Runtime performance)

In this experiment, the runtime performance for each algorithm on different datasets was examined. Runtimes exceeding 24 hours are classified as a *Timeout*. In the case of FocusCO, 5 random nodes were supplied as query nodes.

<i>Dataset</i>	<i>Algorithm</i>	<i>Runtime (seconds)</i>	<i>Runtime (HH:MM:SS)</i>	<i>C</i>
SAPO	CESNA	2393	00h 39m 53s	10
SAPO	CODICIL	<i>Timeout</i>	...	...
SAPO	FocusCo	25955	07h 12m 35s	6
SAPO Hashtags	CESNA	252	00h 04m 12s	10
SAPO Hashtags	CODICIL	5301	01h 28m 21s	12
SAPO Hashtags	FocusCO	82	00h 01m 22s	7
SAPO LIWC	CESNA	0.08	<1s	8
SAPO LIWC	CODICIL	0.4	<1s	2
SAPO LIWC	FocusCO	8	00h 00m 08s	2

Table 5.3: Runtime performance for the different algorithms on different datasets. The best performing algorithm on each dataset is written in bold. *C* denotes the number of communities detected

Looking at Table 5.3, we can see that CESNA is performing quite well for large networks. FocusCO was the fastest when it comes to medium-sized

networks. When it comes to small networks, all algorithms returns a result in near-instant time. FocusCO took quite some time for large networks, but managed to produce some results in contrast to CODICIL. For medium-sized networks FocusCO performed well and managed to produce results within a feasible time. An interesting observation is that the probabilistic community detection CESNA outperformed community search algorithm FocusCO on large scale networks. While CODICIL performs drastically worse on larger instances, the difference between CESNA and FocusCO is hard to conclude due to the difference in implementation quality. Looking at the number of communities detected, we can see that there are some differences. CESNA detects more communities than FocusCO. CESNA also detects more communities than CODICIL for the small network.

5.3 Experiment 2 (Iterative community detection)

In this experiment, we applied our *probabilistic divisive community detection* method on the SAPO Twitter dataset. Our method further clusters the communities that matches a user-provided query, with the goal to find sub-communities with well-connected and attributed similar nodes. The attributes selected to be part of the query was randomly selected. The cosine similarity and density is measured in comparison to the initially generated communities. The average cosine similarity is a measure of the average attributed similarity of the member-nodes, supporting high-dimensional positive spaces. Density is a measure of how densely connected the nodes are with each other, where in a dense network, the number of edges are close to the maximum number of possible edges. The cosine-similarity and density is computed in the following way:

The cosine similarity of two nodes' feature vectors v and u is:

$$\text{cosine}(v, u) = \frac{v \cdot u}{||v||_2 \cdot ||u||_2}$$

And yields a value $[-1, 1]$ where -1 means exactly opposite, and 1 meaning exactly the same, with 0 indicating decorrelation.

The density for a network $G(V, E)$ is:

$$D = \frac{|E|}{|V| \cdot (|V| - 1)}$$

For each iteration of our method, we compare the density and cosine simi-

larity. Intuitively, the average cosine similarity of the member-nodes in each community should increase, as well as the edge-density.

<i>Community</i>	Avg. <i>Cos. Similarity</i> <i>1 Iteration</i>	Avg. <i>Cos. Similarity</i> <i>2 Iterations</i>	Avg. <i>Cos. similarity</i> <i>3 Iterations</i>	Avg. <i>Cos. Similarity</i> <i>4 Iterations</i>
1	0.427	0.442	0.496	0.573
2	0.418	0.458	0.468	0.511
3	0.429	0.449	0.486	0.503
4	0.433	0.478	0.490	0.529
5	0.485	0.481	0.464	0.520
6	0.526	0.463	0.450	0.571
7	0.439	0.501	0.492	0.492
8	0.423	0.437	0.468	0.466
9	0.512	0.431	0.443	0.512
10	0.507	0.528	0.435	0.495

Table 5.4: Comparison of the cosine similarity within the resulting communities

<i>Community</i>	<i>Density</i> <i>1 Iteration</i>	<i>Density</i> <i>2 Iterations</i>	<i>Density</i> <i>3 Iterations</i>	<i>Density</i> <i>4 Iterations</i>
1	0.003	0.030	0.114	0.222
2	0.002	0.031	0.078	0.256
3	0.004	0.030	0.110	0.182
4	0.001	0.016	0.043	0.197
5	0.002	0.011	0.053	0.176
6	0.001	0.021	0.082	0.191
7	0.002	0.016	0.056	0.148
8	0.002	0.016	0.092	0.202
9	0.001	0.018	0.094	0.197
10	0.001	0.016	0.096	0.245

Table 5.5: Comparison of the density within the resulting communities

Looking at the results, we can see that they are mostly in line with our intuition. In 8 out of 10 cases, the average cosine similarity between the member nodes was higher after 4 iterations. However, only 4 out of 10 cases displayed a consecutive improvement for each iteration. It seems that in some cases, further clustering decreases the average cosine similarity. We believe that the reason for this is that CESNA sometimes clusters nodes with a high association to a single common attribute, rather than nodes having a medium-high association to multiple attributes. This results in the nodes being less similar in many attributes, but very alike on a single attribute. The overall similarity for all attributed would thus decrease. Looking at the results, we can also see that, in all cases, the density increases for each iteration. This is not surprising, since nodes not connected to the community are removed, reducing the maximum number of possible edges, which yields a higher density if communities have many edges within the community and sparse outbound edges.

5.4 Case Study: Detecting topic-based communities

The aim of this experiment was to investigate how well communities formed around a topic of interest can be identified. In this experiment, the SAPO-Hashtag dataset was used. Our intuition is that communities formed around a topic, should have members discussing things related to the community. Adding a hashtag to written text, is a common way to assign a topic or context to the written text on social media. CESNA was set to identify 10 communities, followed up by manually analyzing the resulting communities. Table 5.6, displays the resulting communities we believe are formed around a topic based on which attributes (hashtags) had the highest association to the community. The density of edges within each community was examined, as well as the fraction of nodes using each associated hashtag.

C_{ID}	N	D	μ_{CA}	μ_{GA}	A_+	A_-
1	2966	0.048	0.616	0.040	svpol	love
			0.026	0.002	sdgs	seo
			0.148	0.012	sweden	infosec
			0.022	0.001	southsudan	new
			0.036	0.004	humanrights	follow
3	2310	0.051	0.385	0.008	säkpöl	sdgs
			0.270	0.007	migpol	innovation
			0.282	0.005	svfm	love
			0.763	0.040	svpol	new
			0.191	0.004	föpol	seo
5	2648	0.027	0.080	0.007	innovation	ff
			0.092	0.003	sthlmtch	love
			0.066	0.002	swgreen	ifkbg
			0.071	0.003	skola	follow
			0.038	0.001	ehandel	new
6	2783	0.013	0.066	0.002	ifkbg	ff
			0.157	0.008	twittboll	infosec
			0.083	0.003	aik	contentmarketing
			0.058	0.002	bajen	sdgs
			0.036	0.003	lfc	innovation
7	2287	0.013	0.046	0.002	infosec	migpol
			0.029	0.002	sdgs	ifkbg
			0.052	0.007	innovation	expressen
			0.145	0.008	russia	svt
			0.033	0.002	privacy	aftonbladet
network	45752	0.001	—	—	—	—

Table 5.6: A sample of the detected communities. Each row contains the Community ID (C_{ID}), number of nodes who are members of the community (N), the edge density within the community (D), the fraction of nodes in the community that have an association to the attribute (μ_{CA}), the fraction of nodes in the entire network that have an association to the attribute (μ_{GA}), Top 5 attributes with the highest association (A_+), Top 5 attributes having the most negative association (A_-). Note that μ_{CA} and μ_{GA} are for the positively associated attributes.

Looking at the communities in Table 5.6, we argue for the following:

- Community 1 is formed around politics related to Sweden and human rights. The hashtags having the highest association are either related to Sweden, Swedish politics, sustainable development and human rights in foreign countries. The hashtags having negative association is mostly generic hashtags such as "love". Furthermore, infosec (information security) does not seem to be discussed in this community. This is most likely an indication that the political discussions are about non-technical subjects.
- Community 3 is formed around politics related to immigration, national security and defense. This community have a high association to hashtags used for national security politics (säkpol), immigration politics (migpol), national defense politics (föpol), the Swedish Armed Forces (svfm) and Swedish politics in general (svpol). Looking at the hashtags with a negative association, this community does not seem to be interested innovation or sustainable development goals.
- Community 5 is formed around entrepreneurship and innovation. This community seem to be interested in things related to innovation. Discussing things such as the Stockholm tech-scene (sthlmttech) and "green" solutions for Sweden (swgreen), school (skola) and e-shopping (ehandel). Notable attributes with a negative association is "ifkgbg", used for discussing things related to the Swedish football team IFK Göteborg. However, this is most likely a negative association to sports in general and not a specific team.
- Community 6 is formed around sports. Attributes having a high association to the community is a number of sports clubs with successful football teams. In addition to "ifkgbg", we can see that the Swedish teams AIK and Hammarby (also known as Bajen) are present. The English team Liverpool Football Club (lfc) is also highly associated. "twittboll" is also associated, used for discussing football in general. The hashtags having negative association are hashtags used for politics and business innovation.
- Community 7 is formed around privacy and surveillance concerns. This community seem intereseted in talking about information security, privacy, innovation and Russia as well as sustainable development goals. Topics not discussed are things related to the Swedish media, immigration politics and sports.

Communities not listed in Table 5.6 are either formed around very general hashtags such as "love" or "fun", or is a combination of the listed communities.

In this experiment, the 250 most common hashtags of the network was selected as attributes. Incorporating domain knowledge by selecting attributes related to a specific use case could potentially detect communities that are formed around more specific topics than the topics we randomly selected. However, the topic related to each community listed in Table 5.6 is specific enough for many use cases in different domains.

5.4.1 Detecting sub-topics

The next step was to try to see if we could get more fine-grained communities by using our probabilistic divisive method. To examine this, we tried to detect probable sub-communities within a community. It would be probable that sub-communities formed around different sports or teams exists within the community formed around sports (Community 6).

C_{ID}	N	D	μ_{CA}	μ_{GA}	A_+	A_-
6.4	244	0.118	0.038	0.009	twittpuck	ifkgbg
			0.049	0.002	nhl	twittboll
			0.206	0.004	shl	brexit
			0.066	0.001	hv71	svt
			0.045	0.001	fbkse	aik
6.9	173	0.136	0.587	0.002	ifkgbg	pjnet
			0.116	0.003	gbgftw	tcot
			0.011	0.004	superbowl	brexit
			0.011	0.004	media	refugees
			0.029	0.005	nyttjobb	photography
6.10	138	0.152	0.679	0.003	aik	ifkgbg
			0.007	0.002	nigeria	brexit
			0.095	0.007	stockholm	trump
			0.058	0.001	hundralappen	periscope
			0.102	0.012	sweden	svt
network	2783	0.013	—	—	—	—

Table 5.7: A selection of the resulting sub-communities, generated by iteratively applying CESNA on the sports-community. Each row contains the Community ID (C_{ID}), number of nodes who are members of the community (N), the edge density within the community (D), the fraction of nodes in the community that have an association to the attribute (μ_{CA}), the fraction of nodes in the entire network that have an association to the attribute (μ_{GA}), Top 5 attributes with the highest association (A_+), Top 5 attributes having the most negative association (A_-). Note that μ_{CA} and μ_{GA} are for the positively associated attributes.

In Table 5.7 a selection of the detected sub-communities are displayed. We find three sub-communities corresponding to different sports or teams. Community 6.4 have high association to topics related to hockey. The topic with the highest association is "shl" which is a hashtag for the Swedish hockey league. The other hashtags are names of Swedish hockey teams (hv71 and fbkse), as well as the name of the American hockey league "nhl" and the general hockey hashtag "twittpuck". This community has a negative association to soccer related hashtags such as "ifkgbg", "aik" and "twittboll". Community 6.9 have a high association to the topics "ifkgbg", and

"gbgftw". Both of these are used when discussing the team IFK Göteborg. Community 6.10 has a high association to the hashtag "aik", representing the team AIK, and a negative association to the team IFK Göteborg. The interpretation of these results indicates that community 6.4 is formed around hockey, community 6.9 is formed around the team IFK Göteborg, and community 6.10 is formed around the team AIK.

5.4.2 Community search

The next step was to use FocusCO, with a well defined query set, and see how the resulting communities compares to the ones identified by our iterative approach. The sports community (community 6 in Table 5.6) was used as the input network. The query-set of nodes, was selected using our top-sampled community search method, yielding the 5 users with the highest association to the hashtag "ifkbg". Our intuition is that we should get communities associated to the team IFK Göteborg, similar to community 6.9 in Table 5.7. The resulting communities can be seen in Table 5.8

C_{ID}	N	D	μ_{CA}	μ_{GA}	A_+
CS1	2	0.000	1.000	0.040	svpol
			0.5	0.010	melfest
			0.5	0.009	twittpuck
			0.5	0.007	stockholm
			0.5	0.005	isis
CS2	2	0.000	1.000	0.002	ifkbg
			0.5	0.008	twittboll
			0.5	0.007	migpol
			0.5	0.003	humanrights
			0.5	0.004	refugees
CS3	151	0.093	1.000	0.002	ifkbg
			0.331	0.008	twittboll
			0.272	0.040	svpol
			0.166	0.009	twittpuck
			0.132	0.003	aik
CS4	2625	0.014	0.267	0.040	svpol
			0.147	0.008	twittboll
			0.126	0.009	twittpuck
			0.098	0.010	melfest
			0.080	0.003	aik
network	2783	0.013	—	—	—

Table 5.8: Resulting communities for applying FocusCO on the sports-community, using the 5 most frequent users of the hashtag "ifkbg" as query-nodes. Each row contains the Community ID (C_{ID}), number of nodes who are members of the community (N), the edge density within the community (D), the fraction of nodes in the community that have an association to the attribute (μ_{CA}), the fraction of nodes in the entire Twitter network that have an association to the attribute (μ_{GA}), Top 5 attributes with the highest association (A_+).

Looking at Table 5.8, we can see that it yields four communities. Two of these communities (community CS1 and CS2) only have 2 member nodes, community CS4 contains the majority of nodes in the sub-network, and community CS3 which is about the same size as community 6.9 in Table 5.7.

Community CS3 also has a high association to hashtags related to the team IFK Göteborg. In fact, all nodes in community CS3 have been using the hashtag "ifkgbg" in contrast to the number of nodes in community 6.9 which was 58.7%. On the other hand, the density of community 6.9 (0.136) is higher than the density of community CS3 (0.093). Both community 6.9 yielded by our iterative approach and community CS3 yielded by our top-sampled community search approach, seems to detect the sub-communities we aimed at. However, there is a trade-off in density versus queried attribute association.

5.5 Discussion

The first experiment showed that both CESNA and FocusCO are good candidates for large scale analysis. The probabilistic approach of CESNA results in great scalability as the network grows. FocusCO also performed quite well as the network grew in size. The runtime on large networks was worse in the case of FocusCO, but it outperformed CESNA on a medium-sized network with relatively many attributes. CODICIL only performed well in the case of a small network. We consider both CESNA and FocusCO to be good candidate algorithms for analyzing communities in large scale node-attributed networks.

The second experiment showed how the probabilistic divisive community detection method (in most cases) can be used to get less general similarities within the clusters as well as increasing the density within the clusters. When using our proposed method, the selection of attributes is quite important and may yield different results depending on the use-case. However, the numbers seem to suggest that when further clustering the communities with respect to a user-query, well defined sub-communities were identified.

The last experiment illustrated, through a case study, how a topic of interest seems to be associated to most communities. Using the approach we did, it would have been expected for most of the communities to be associated to very common, but generic hashtags/topics such as "love", "fun" or "ff". The experiment shows however that the hashtags/topics associated to each community follow some sort of theme, or are all related to a single, more broad topic (i.e sports). Using our presented divisive community detection method, we were able to get the sub-themes of a community's theme (i.e what sport team they discuss). We also showed that FocusCO was able to retrieve a sub-theme community, using our top-sample method for automatically generating exemplar nodes. Comparing the quality of the

two sub-theme communities formed around the same sub-theme, they both performed quite well, but our iterative approach had a slight edge in density, while FocusCO had an edge in the fraction of nodes directly discussing the sub-theme. This experiment shows how the user preferences "find communities formed around sports" or "find communities formed around a specific team" can be incorporated into the analysis. This could further be improved by instead of selecting the 250 most common hashtags, only select attributes relevant for the specific domain.

6 Psychological profiling in social networks

Some of the attributes describing a user in an online social network is not directly available on the user’s profile. While the profile may contain location, language, and account activity, they are most likely not sufficient for describing the user’s personality. To infer personality-related attributes, text written by the user can be analyzed and translated into psychographic traits. Looking at the results from the previous chapter, we believe that it would be possible using node-attributes correlating to each user’s personality to detect communities formed around psychographically like-minded people. This chapter will describe how our proposed community detection methods could be used to detect such communities.

6.1 Linguistic and psychographic features in social networks

In this thesis we define psychographic features as features that relate to an individual’s personality, attitudes, opinions, interests and lifestyle. In real-life social networks, psychographic attributes are rarely available unless there is a deep knowledge of the individuals in the network. Recent research has shown examples of how psychographic features can be derived from the data that is available in online social networks using information about the users online behavior and their writing style [37, 4, 5, 3]. Some psychographic features can be determined by analyzing linguistic features. For example, linguistic features has been used to identify people with traits of a radicalized mindset [3]. Psychographic features can be of great interest for intelligence analysts, trying for example to detect communities of radicalized people.

6.2 Related research

Node attributes available on social networks are often limited. While demographic attributes such as gender, age, location tend to be available, psychographic attributes, such as the user’s level of anger or openness, is not. Methods have been developed to derive these attributes from data available on social networks. Golbeck *et al.* [4] examined the correlation between linguistic features in tweets, and scores for the *big five personality model*. Kosinski *et al.* [5] analyzed Twitter profiles of 335 users for which they had gathered scores related to the big five personality model. Kosinski

et al. finds interesting correlations between the big five personality model and attributes for the Twitter profiles, such as: the number of followers, number of people they follow and number of times they’ve tweeted. Psychographic features that relates to radicalization was identified by Kaati *et al.* [3]. They analyzed a Swedish discussion forum and in particular a group of users on the forum that they refer to as extreme adopters. The extreme adopters were found to have certain linguistic features indicating that they have a ”radicalized mind-set”.

6.3 LIWC

Linguistic Inquiry and Word Count (LIWC)[38] is a text analysis tool for performing psychological groupings of linguistic features. LIWC uses a reference dictionary of words and their relation to a psychological group. LIWC is a well adopted method that has been employed in numerous studies with the aim to study psychological traits based on written text [4, 3, 37]. LIWC has 81 official psychological categories such as anger, exclusion, certainty etc. Each category will get a score depending on the number of pre-defined words or n-grams associated to the category, that are present in the input text. This score is typically normalized by the total number of words in the input text and compared to a baseline. This way, you can see if the input text is more or less angry/exclusive/etc. than the baseline linguistics. The linguistic features derived from LIWC have been studied in correlation to many personal traits. Kaati *et al.* [3] examined the linguistic markers of forum users in correlation to a ”radicalized mind-set”. Golbeck *et al.* [4] examined the correlation between the linguistic markers of Twitter users and the corresponding user’s personality. Performing a LIWC analysis on written text is typically done in the following way: for each psychological category you want to study, you’ll need a word list with words related to the category. You then count all the category-related words (for all categories) in the written text and normalize it by the total number of words. This normalized count is then compared to a baseline. For each category the baseline may be higher or lower than the normalized count. This set of differences is typically the interesting metric to study.

Big five personality traits The big five personality traits are known as the *five factor model (FFM)*. The five factor model is one of the most well-researched and well-regarded measures of personality in recent years [4]. The model describes a personality using five traits.

- Openness to experience (*curious vs. cautious*)

- Conscientiousness (*organized vs. careless*)
- Extraversion (*outgoing vs. solitary*)
- Agreeableness (*cooperative vs. antagonistic*)
- Neuroticism (*nervous vs. confident*)

Golbeck *et al.* [4] tried to predict the personality of Twitter users by applying LIWC and trying to find a correlation between the LIWC scores and the big five personality traits. The authors was able to predict these personality traits in a 10-step interval with a decent accuracy. While this is not ideal, it is most likely sufficient for many applications. The authors discuss the potential of future use of these scores and suggests that it could be used to connect like-minded people. In 2015, Alahmadi *et al.* [37] clustered the websites in the outgoing Internet traffic in order to examine how the content relates to psychological constructs such as the big five personality traits of the person browsing the content. Alahmadi *et al.* demonstrated how predictive assessment can be made between the content available on a website and the associated personality traits. This could potentially be used to improve the prospects of preventing insider attacks within an organization.

6.4 Inferring psychographic attributes

In order to infer psychographic attributes for the nodes in the network, one could apply a LIWC analysis. For a Twitter network, LIWC scores can be assigned in the following way: for each user in the network, go through their tweets and count all the occurrences of words related to the LIWC categories described by Kaati *et al.* [3]. This way, each LIWC category gets a count. This count is later normalized by the total number of words in all of their tweets. The normalized count is then compared to a baseline. The next step is to discretize the difference into ranges. For each category, a user can be assigned 1 out of 5 values. *Very low* for being more than 3 standard deviations lower from the baseline. *Low* for being 1-3 standard deviations lower than the baseline. *Average* for being within one standard deviation from the baseline. *High* for being 1-3 standard deviations higher than the baseline. And *Very high* for being more than 3 standard deviations higher than the baseline. This results in a total of 5 attributes per LIWC category.

We are optimistic that communities with similar psychographic traits can be identified. Communities with similar psychographic traits in conjunction with what topics they tend to discuss, would be of great interest for FOI and other research institutions or private actors. Not only would our methods

be able to tell the user what topics they discuss, they might also be able to tell whether they are hostile etc.

7 Conclusion

The algorithms and methods discussed in this thesis are difficult to evaluate in terms of accuracy. This is largely due to the nature of communities. Community detection algorithms for node-attributed networks makes a trade-off where the network's structural properties is taken less into account, in favor for incorporating attributed similarity. This leads to the resulting communities having a lower score in categories such as flow, density or modularity. While loosening up the definition of communities as densely connected sub-networks, these algorithms may result in better communities for the specific use case.

Comparing the resulting communities to "known communities" is also difficult. Communities can be assembled based on different characteristics where no assembled community is necessarily more "true" than the other. However, some characteristics are arguably more "natural" in certain networks than others. Similar topics of interest in combination with structural similarities, seems to be a natural characteristic for social networks. Community defining characteristics that are not "natural" in social networks could be the number of written tweets/posts. Different datasets each have their own type of "natural" communities and the answer to what characteristics are "natural" is highly dependent on the aim of the analysis. Different algorithms yield communities of different characteristics. Thus, the algorithm should be chosen that detects communities of the desired characteristics and the resulting communities should be interpreted according to the domain and characteristics of the analysis. Methods such as CESNA yields more interpretable results in the form of communities and weights for the associated attributes. The ease of interpretation is a important property for analyzing the characteristics in these types of networks. Social constructs such as communities are complex and manual evaluation of different characteristics is most likely the best method to understand the nature of that community.

When working with node-attributed networks, it is important to be careful when introducing a new attribute to the network. Any attribute present in the dataset may be the factor of which a group of nodes is grouped on. Attributes that are not of interest and/or should not have any affect on the resulting communities should thus be omitted. In the experiments for detecting topic-based communities, most of the communities have an association

to broad topics such as "love", "fun" or Swedish politics (svpol). These topics are very broad and does not say anything by themselves. Broad topics like these could have been omitted in favor for more specific topics. Alternatively, these topics could have been omitted in favor for a smaller attribute space, resulting in a faster runtime.

The aim of this thesis was to investigate methods for feasible community detection within node-attributed networks. Furthermore, we aimed to investigate what types of communities could be detected and if high-level user preferences could steer the analysis. The proposed methods were able to cluster a large node-attributed social network in a feasible time. The probabilistic divisive clustering method and the top-sampled community search method both have their way of letting the user steer the analysis according to their preferences. The experiments, displays that well defined communities and their sub-communities can be identified by either using the presented probabilistic divisive community detection method as well as by using the top-sampled community search method. The proposed methods were able to identify communities and sub-communities, formed around different topics of interest. We believe that these methods are potentially usable for user-oriented large scale analysis and should further be investigated.

7.1 Future Work

While this thesis demonstrates the potential of community detection in large node-attributed networks, there are many opportunities for extending the scope of this thesis. This section presents three of these directions.

7.1.1 Extending the node-attributes

One possible extension would be to extend the community detection methods, so that they can handle more general types of attributes. For example, allowing for non-binary valued attributes. This could be useful for attributes that has some form of weight. Topics discussed could for example be a weight depending on how much a user is discussing it.

7.1.2 Other data structures

In this thesis, we have only looked at node-attributed networks. Extending and applying these methods to temporal networks and/or edge-attributed networks would be one possible future direction. Comparing the methods discussed in this thesis to methods for multi-layer or multiplex networks would be another possible direction.

7.1.3 Other use cases

In this thesis we mainly looked at topic-based communities. However, applying these methods to networks with attributes for different use-cases would be interesting. As we discussed in the previous chapter, assigning psychographic attributes to the users in a network and investigating the resulting communities would be of great interest. It would also be interesting to apply these methods in non-social networks, such as biological networks.

Bibliography

- [1] J. Yang, J. J. McAuley, and J. Leskovec, “Community detection in networks with node attributes,” *CoRR*, vol. abs/1401.7267, 2014.
- [2] J. Yang and J. Leskovec, “Community-affiliation graph model for overlapping network community detection,” in *Proceedings of the 2012 IEEE 12th International Conference on Data Mining, ICDM '12*, (Washington, DC, USA), pp. 1170–1175, IEEE Computer Society, 2012.
- [3] L. Kaati, K. Cohen, T. Isbister, and A. Shrestha, “Linguistic markers of a radicalized mind-set among extreme adopters,” *1st International Workshop on Cyber Deviance Detection (CyberDD)*, 2017.
- [4] J. Golbeck, C. Robles, M. Edmondson, and K. Turner, “Predicting personality from twitter.,” pp. 149–156, 2011.
- [5] D. Quercia, M. Kosinski, D. Stillwell, and J. Crowcroft, “Our twitter profiles, our selves: Predicting personality with twitter,” *IEEE International Conference on Social Computing*, 2011.
- [6] N. Mishra, R. Schreiber, I. Stanton, and R. E. Tarjan, “Finding strongly knit clusters in social networks,” *Internet Mathematics*, Internet Mathematics.
- [7] X. Jierui and K. S. Boleslaw, “Towards linear time overlapping community detection in social networks,” *Advances in Knowledge Discovery and Data Mining*, 2012.
- [8] J. McAuley and J. Leskovec, “Learning to discover social circles in ego-networks,” *NIPS '12*, 2012.
- [9] Y. Ruan, D. Fuhry, and S. Parthasarathy, “Efficient community detection in large networks using content and links,” pp. 1089–1098, 2013.
- [10] R. Balasubramanyan and W. W. Cohen, “Block-lda: Jointly modeling entity-annotated text and entity-entity links.,” *SDM '11*, 2011.

- [11] C. Bothorel, J. D. Cruz, M. Magnani, and B. Micenková, “Clustering attributed graphs: models, measures and methods,” *CoRR*, vol. abs/1501.01676, 2015.
- [12] Y. Fang, R. Cheng, X. Li, S. Luo, and J. Hu, “Effective community search over large spatial graphs,” *Proc. VLDB Endow.*, 2017.
- [13] B. Perozzi, L. Akoglu, P. Iglesias Sánchez, and E. Müller, “Focused clustering and outlier detection in large attributed graphs,” 2014.
- [14] S. Fortunato and D. Hric, “Community detection in networks: A user guide,” *Physics Reports*, 2016.
- [15] N. P. Nguyen, T. N. Dinh, D. T. Nguyen, and M. T. Thai, “Overlapping community structures and their detection on social networks,” *SocialCom/PASSAT*, 2011.
- [16] S. Fortunato, “Community detection in graphs,” *Physics Reports*, 2010.
- [17] L. Rokach and O. Maimon, “Clustering methods,” *Data Mining and Knowledge Discovery Handbook*, 2005.
- [18] D. M. Blei, A. Ng, and M. Jordan, “Latent dirichlet allocation,” *JMLR*, 2003.
- [19] S. Johnson, “Hierarchical clustering schemes,” *Psychometrika*, 1967.
- [20] S. Fortunato, “Community detection in graphs,” *Physics Reports*, 2010.
- [21] J. Xie, S. Kelley, and B. K. Szymanski, “Overlapping community detection in networks: the state of the art and comparative study,” *ACM Computing Surveys*, 2013.
- [22] M. Ester, R. Ge, B. Gao, Z. Hu, and B. Ben-Moshe, “Joint cluster analysis of attribute data and relationship data: the connected k-center problem,” *SDM '06*, 2006.
- [23] F. Moser, R. Colak, A. Raey, and M. Ester, “Mining cohesive patterns from graphs with feature vectors,” *SDM '09*, 2009.
- [24] Y. Ruan, D. Fuhry, and S. Parthasarathy, “Efficient community detection in large networks using content and links,” *WWW '13*, 2013.

- [25] Z. Yang, H. Cheng, and J. Yu, “Graph clustering based on structural/attribute similarities,” *VLDB '09*, 2009.
- [26] R. Balasubramanyan and W. W. Cohen, “Block-lda: Jointly modeling entity-annotated text and entity-entity links.,” *SDM '11*, 2011.
- [27] Y. Liu, A. Niculescu-Mizil, and W. Gryc, “Topic-link lda: joint models of topic and author community,” *ICML '09*, 2009.
- [28] Z. Xu, Y. Ke, Y. Wang, H. Cheng, and J. Cheng, “A model-based approach to attributed graph clustering,” *SIGMOD '12*, 2012.
- [29] Y. Sun, C. Aggarwal, and J. Han, “Relation strength-aware clustering of heterogeneous information networks with incomplete attributes,” *VLDB '12*, 2012.
- [30] S. Fortunato and A. Lancichinetti, “A comparative analysis of community detection algorithms on artificial networks,” 2009.
- [31] Z. Yang, R. Algesheimer, and C. J. Tessone, “Community detection algorithms: A comparative analysis: Invited presentation, extended abstract,” *Scientific Reports*, 2016.
- [32] J. Yang and J. Leskovec, “Overlapping community detection at scale: A non-negative factorization approach.,” *WSDM '13*, 2013.
- [33] W. Cui, Y. Xiao, H. Wang, Y. Lu, and W. Wang, “Online search of overlapping communities,” *SIGMOD '13*, 2013.
- [34] Q. Niu, P. Lai, S. M. Faisal, S. Parthasarathy, and P. Sadayappan, “A fast implementation of MLR-MCL algorithm on multi-core processors,” pp. 1–10, 2014.
- [35] S. van Dongen, “Graph clustering by flow simulation,” 2000.
- [36] L. Townsend and C. Wallace, “Social media research: A guide to ethics,” 2016.
- [37] B. A. Alahmadi, P. A. Legg, and J. R. C. Nurse, “Using internet activity profiling for insider-threat detection,” in *17th International Conference on Enterprise Information Systems (ICEIS)*, pp. 709–720, 2015.

- [38] Y. R. Tausczik and J. W. Pennebaker, “The psychological meaning of words: Liwc and computerized text analysis methods,” *Journal of Language and Social Psychology*, 2010.