Cartography M.Sc.

Master thesis

A Comprehensive Study on Bike Sharing Mobility in the City of Munich: Utilizing Community Detection Method

Yağız Kara

2023

A Comprehensive Study on Bike Sharing Mobility in the City of Munich: Utilizing Community Detection Method

submitted for the academic degree of Master of Science (M.Sc.)
conducted at the Department of Aerospace and Geodesy
Technical University of Munich

Author: Yağız Kara
Study course: Cartography M.Sc.
Supervisor: MSc. Peng Luo (TUM)
Co-Supervisor: MSc. Sebastian Seisenberger (TUM)

Reviewer: Dr. Eva Hauthal (TUD)

Chair of the Thesis
Assessment Board: Prof. Dr. Liqiu Meng

Date of submission: 07.09.2023

Statement of Authorship

Herewith I declare that I am the sole author of the submitted Master's thesis entitled:

"A Comprehensive Study on Bike Sharing Mobility in the City of Munich: Utilizing Community Detection Method"

I have fully referenced the ideas and work of others, whether published or unpublished. Literal or analogous citations are clearly marked as such.

Munich, 07.09.2023 Yağız Kara

Acknowledgment

First of all, I would like to express my gratitude to my supervisors, Peng Luo and Sebastian Seisenber, who have provided valuable suggestions and been supportive during the whole research.

I would like to extend my sincere thanks to the Chair and all the staff of Cartography M.Sc. for helping their students in many cases and providing unlimited materials for us to work with.

Finally, I would like to express my deepest appreciation to all the people who guided me along this incredible journey. I must mention especially my dear parents & sister, Seher, İsmail, and Deniz; and my sweetheart, Ece. They gave me support, motivation, and love to successfully finish my studies. Nothing would be possible without them.

Sizleri çok seviyorum.

Abstract

This study focuses on contemporary bike-sharing systems, specifically hybrid systems that blend docked and dockless approaches. These systems offer sustainable urban transportation solutions, but research has mainly concentrated on station-based and dockless variants, leaving hybrid systems underexplored. Our research introduces a spatiotemporal analysis workflow centered on dynamic community detection within Munich's hybrid bike-sharing network. The primary objective is to understand user travel behavior and its connection to the urban environment. We develop this workflow, assess existing analysis methods, adopt dynamic community detection algorithms, and apply the workflow to urban transport planning. Our research addresses various research questions, exploring spatial patterns, temporal considerations, network analysis attributes, semantic information, and the advantages of dynamic community detection. Key findings highlight the dynamic nature of urban mobility influenced by temporal factors and the impact of built environment features on travel patterns. Land use information assigned to communities reveals distinct travel purposes among them. While this research demonstrates the potential of dynamic community detection, it acknowledges computational challenges and suggests future work in GI Science. In conclusion, our methodology contributes to Geographic Information Science, Cartography, and Urban Transport Planning, offering insights for optimizing bike-sharing systems and promoting sustainable urban transportation.

Keywords: Dynamic Community Detection, Human Mobility, Bike Sharing

Contents

List of Figures

List of Tables

Equations

List of Abbreviations

BS: Bike-Sharing

OD: Origin-Destination

GPS: Global Positioning Systems

MVG: Münchner Verkehrsgesellschaft GmbH

D/S: Demand/Supply

CSV: Comma Seperated Value

ÄDBV: Office for digitization, broadband and surveying Munich (Amt für Digitalisierung, Breitband und Vermessung München)

ALKIS: Authoritative Real Estate Cadastre Information System

Nomenclator

G : OD Graph. Consists of a set of Graphs for each time interval.

C^t : Communities in specific t^{th} interval of graph G^t.

C_i^t: i^{th} community in specific time interval communities.

$G^C = (V^C, E^C)$: Consensus graph/network—> Constructed from “community membership graphs” which have undirected weighted edge connecting nodes of the same community.

V^C : Union of all nodes within all snapshots.

E^C : Unions of all edges in all community membership graphs.

w_{ij}^c : The weight of an edge in the consensus graph, represents the time duration of an edge. Shows how many times nodes stay connected within a community among all snapshots.

τ: total number of snapshots.

C^C: Set of consensus communities within the consensus graph G^C.

C_i^C : i^{th} consensus community in consensus communities C^C within consensus graph G^C.

$C^{D,t}$: set of Dynamic communities in a snapshot G^t (where C_i^C and $C_i^{D,t}$ have the same set of nodes.)

$SC_i^{D,t}$: set of dynamic sub-communities.

1. Introduction

Bike-sharing systems have evolved from the first pioneering systems in Amsterdam in 1965 and second-generation, coin-based systems in Copenhagen in the 1990s to today's fourth-generation, hybrid systems (Z. Chen et al., 2020; Eren & Uz, 2020; Liao et al., 2023; Shaheen et al., 2010; L. Zhang et al., 2015). The hybrid bike-sharing systems are a combination of docked and dockless systems, allowing its users to park their bicycles independently from a station, anywhere in the operation area, and the extent of the bicycle fleet is not limited to the station size (Kou & Cai, 2021; Shen et al., 2018; Shui & Szeto, 2020). These systems have attained popularity with their advantages such as deflating traffic intensity during peak hours, lowering carbon emissions, service to human health, decreasing car ownership, and having an essential role in sustainable transportation discussions (Song et al., 2021).

Although public transport ensures an extensive mobility function, it is not possible for them to reach all parts of the urban regions as vehicles do (Fan et al., 2019). Due to being an eligible way of city traveling, shared bike systems are an effective way to link the first or last mile of the journey with public transportation systems and a beneficial tool for complimentary mode in public transportation (Liao et al., 2023; Lu et al., 2023; Shui & Szeto, 2020). Constant upgrowth in bicycle-sharing systems made researchers focus on bike-sharing and has crucially increased publications in the literature (Fishman, 2016; Shui & Szeto, 2020; Song et al., 2021). Up to the author’s knowledge, there are a great number of studies regarding station-based and dockless bike-share systems but, studies related to hybrid bike-sharing systems are limited and need further investigation (Albiński et al., 2018; Kou & Cai, 2021).

The latest generation of shared bikes is hybrid systems with GPS devices embedded. This allows the unique gathering of **large-scale travel data** which enables scientists to study user's **travel behavior** and examine **built environment** effects on the system level (Z. Chen et al., 2020). Discovering, and analyzing GPS data on bike-share usage innovatively with **spatiotemporal analysis** methods ensures decision support for service enhancement (Lu et al., 2023). Therefore, it is necessary to apply novel spatiotemporal analysis methods to reveal users’ travel behavior patterns and their relation to built environment factors.

Mobility data can be visualized as origin/destination pairs as points, and straight lines in between these pairs as travel flows. Visualizing travel flows can allow us to understand urban mobility patterns and analyze the reasonings behind them. However, human mobility has a dynamic structure and shows complex patterns throughout different times of the day, different days of the week, or different seasons of the year. Additionally, with the help of new GPS systems, it is possible to collect hundreds of thousands of trip data that are generated by humans which also makes this data complex to evaluate. At this point, we can benefit from Community

Detection Methods of Network Science to address revealing meaningful travel patterns by converting bike share traffic data into a mobility network. The community detection method determines clusters or modules that have more intra-region journeys than inter-region journeys within bike-sharing mobility data. The goal of such algorithms is to understand spatial patterns of bike share traffic from a network viewpoint and validate spatial pattern analysis methods(Song et al., 2021).

Community detection methods are efficient in finding static communities but, they neglect the dynamic characteristics of human mobility patterns. Static community detection approaches that use specific periods (snapshots) cannot detect human mobility features entirely. Previous methods are effective in determining progressing communities with great similarity among two subsequent periods (Zhao et al., 2023). Several studies have already applied community detection in urban science, particularly within bike-sharing systems. However, due to the high dynamics of the bike-sharing system (e.g., bike deployment), a dynamic community detection approach is necessary. In this study, we propose a dynamic community detection model and apply it to urban planning as one potential application. In our study, the nodes in the mobility network are basic urban units in Munich, and the links of nodes are flows of bikes.

1.1. Research Identification

This section will introduce the research objective that underpins the research motivation, along with its sub-objectives. Additionally, the research questions devised to achieve these objectives will be presented. Subsequently, the thesis's contribution to the fields of GI Science and Cartography will be discussed.

1.1.1. Research Objectives & Sub-Objectives

This thesis will utilize open-source Origin & Destination GPS data provided by MVG Rad (*MVG Rad*, 2023). The primary objective is to develop a spatiotemporal analysis workflow that models dynamic community structures. This approach aims to enhance the understanding of users' travel behavior within the hybrid bike share system's network and its correlation with the built environment across the city of Munich.

The main objective and its sub-objectives are described as follows;

RO1: Develop a spatiotemporal analysis workflow to model dynamic community structures of hybrid bike share usage.

RO1.1: Evaluate existing spatiotemporal analysis methods that are used to extract bike share travel behavior.

RO1.2: Adopt novel network analysis methods (dynamic community detection methods) to extract cluster/community structures.

RO1.3: Implement and evaluate the proposed workflow as an application of Urban Transport Planning and analyze travel patterns of hybrid bike share usage.

1.1.2. Research Questions

The main objective and its sub-objectives are met by providing solutions to the following RQs, and each of them is enumerated according to corresponding Sub-Objectives (i.e. **RO 1.3 → RQ 1.3.1, RQ 1.3.2**).

RQ 1.1.1. What are the current spatial pattern analysis methods of hybrid bike share system usage?

RQ 1.2.1. What would be the most suitable temporal time unit for the adopted method?

RQ 1.2.2. What additional attributes or parameters could be used to extract meaningful information in a network?

RQ 1.2.3. How would semantic information be assigned to communities?

RQ 1.3.1. Can community detection methods help to extract travel purposes?

RQ 1.3.2. Is there any benefit of defining changing communities?

1.1.3. Detection of Dynamic Communities

The innovation of this research is to generate spatiotemporal analysis workflow by adopting the Dynamic Community Detection Method to reveal community shifts over time in an urban space based on shared bicycle mobility data. To the author's knowledge, there are much research(Lin et al., 2020; Lu et al., 2023; Song et al., 2021) exists regarding community detection of bike-sharing data. However, Zhao et al. (2023) proposed a novel method for detecting dynamic community structures generated based on daily dynamics of human mobility by classifying evolving patterns of urban structure. The dynamic community detection method and its applications need further investigation (Zhao et al., 2023) and the proposed methods will be used for the first time in the city of Munich.

1.2. Thesis Outline

The thesis is structured into six chapters that follow a logical flow. Its focus is on the adaptation of community detection, a method within the field of Network Science, to the discipline of Cartography & GI Science. Furthermore, a methodological framework

is proposed that can be utilized in Urban Transport Planning as a means of analyzing patterns in human mobility.

The thesis starts with Chapter 1, Introduction, which outlines the author's motivation for the implementation of dynamic community detection for bike-sharing mobility networks and the research goal. It also formulates Research objectives & questions and explains the project's contribution to the Cartography and Urban Transport Planning domains.

In Chapter 2, Background and related work, core definitions and fundamental concepts in Bike Sharing Systems, and Community Detection Algorithms are introduced. Moreover, similar research where community detection methods are applied to bike-sharing systems is discussed to address the potential applications of the proposed methods.

In Chapter 3 Methodology we have outlined the research methodology employed in the study, including data preprocessing, static community detection, dynamic community detection throughout the day, and the quality criteria used for analysis.

Following Chapter 4, Case Study, a specific case study is presented. We discussed the test data and the study area, which includes the MVG Rad 2022 Bike-Share Mobility Data (*MVG Rad*, 2023). We applied the proposed workflow to this data, therefore this section involves dynamic community detection within this dataset, involving data preprocessing, static community detection on snapshots, dynamic community detection throughout the day, and dynamic Sub-Community detection.

After the case study section, in chapter 5, we presented the findings and discussions of the research. We also include the outcomes of the community detection analysis and insights derived from the data. Any limitations or constraints may also be discussed in this section.

The final chapter summarizes the study's main conclusions and provides an outlook for future research directions.

2. Theoretical Background & Related Work

Developing a spatiotemporal analysis workflow to model dynamic communities of bike-sharing systems is a complex objective to achieve. A collaboration between various domains such as Urban Trasport Planning, Cartography, GI Science, and Graph Theories is necessary. Therefore, this section provides background information and defines underlying concepts. First, the chapter starts with a general perspective on bike-sharing systems by explaining their emergence, historical background, and system types, along with their significance in the context of Urban Transport Planning. Afterward, to gain an understanding of bike-sharing systems, users' travel behaviors, and spatiotemporal analysis to reveal travel patterns are discussed. In the next section, community detection and its foundation, graph theory, is explained by providing definitions and mathematical expressions. Since the abstraction of real-world network systems such as bike-sharing mobility networks, is necessary for community detection algorithms, the next section introduces how real-world networks are represented. Lastly, the section narrows down to community detection applications on bike-sharing systems.

2.1. Bike – Sharing Systems

2.1.1. Background

In recent years, the need for sustainable transportation plans such as clean fuels, transport technologies, and demand management worldwide became more crucial for experts because of issues such as global climate change, energy security, and nonstable fuel prices (Z. Chen et al., 2020; Eren & Uz, 2020; Liao et al., 2023; Shaheen et al., 2010; L. Zhang et al., 2015). Urban planning and transportation specialists are encouraging diverse eco-friendly travel options, like utilizing public transportation, walking, and biking, to serve as cost-effective means of transportation. This aims to mitigate the adverse impacts associated with extensive car utilization (Bachand-Marleau et al., 2012). In their conventional state, alternative modes of transportation are not as flexible and convenient as cars (Bachand-Marleau et al., 2012; Shaheen et al., 2010). Even though public transport covers long distances, it doesn't expand citywide like automobiles, resulting in the first/last mile problem because of limited connectivity (Fan et al., 2019). The first/last mile problem pertains to challenges arising from the urban and social environment, such as land use mix, neighborhood design, employment opportunities, and the presence of public transport services during the first and final section of a journey (Fan et al., 2019).

Creative solutions have been designed to increase the competitiveness of public transportation against automobiles. Adoption of Bike-sharing programs as a mobility strategy, which involves a shared use of bicycle fleet, emerges as a potential mobility solution to tackle these challenges and promote sustainable urban environments

(Bachand-Marleau et al., 2012; Shaheen et al., 2010). For example, according to research in Shanghai, it was found that bike-sharing usage resulted in a fuel saving of 8358 tonnes which contributed to a reduction in the release of harmful gases, subsequently enhancing the quality of air (Eren & Uz, 2020). Moreover, Shared bicycle systems, offer a promising solution for short-distance trips and address the first/last mile problem. In addition, bike-sharing systems represent a mode of physical activity that can be seamlessly integrated into everyday routines, offers cardiovascular advantages for individuals, reducing the travel times and costs of their users (Bachand-Marleau et al., 2012; Bauman et al., 2017; Eren & Uz, 2020; Fishman, 2016). Thus, many cities have recognized community bike sharing as an efficient method for augmenting urban transportation systems and contributing to urban sustainability (Z. Chen et al., 2020; Eren & Uz, 2020; Lin et al., 2020; L. Zhang et al., 2015). Shaheen et al. (2010) define the concept behind bike-sharing is straightforward. People use bicycles when they need mobility, without the expenses and responsibilities of owning a bike. Bike-sharing allows temporary access to bicycles, as an eco-friendly option for public transportation for its users and this temporary access to the system focuses on daily mobility, enabling individuals to conveniently benefit from public bikes.

Historical Evolution of Bike Sharing

During the 1960s, the concept of "bike-sharing" emerged as a response to growing interest in bicycle usage. This led to the rapid expansion of Bike Sharing Programs across European cities. The initial attempt at bike-sharing, known as the 1st generation, was seen with "White Bikes" in Amsterdam in 1965. Users can take a bike, ride it to an intended destination, and leave it for the other users. Unfortunately, this system was short-lived because of issues like vandalism and theft. The 2nd generation, coin-deposit-based systems, was born in Denmark's Nakskov and Copenhagen in 1993 and 1995, respectively. While being more structured than the previous generation, with docks and a nonprofit system to operate the program, the bikes again experienced stealing because of the anonymity of the rider. This period emphasized the necessity for enhanced bicycle tracking, which became a core concentration for the innovations in 3rd generation bike-sharing programs (Z. Chen et al., 2020; DeMaio, 2009; El-Assi et al., 2017; Eren & Uz, 2020; Fishman, 2016; Shaheen et al., 2010).

The 3rd generation of BSPs, characterized by telecommunication systems, smartphone access, smart cards, kiosks, and computer-aided systems, gained prominence in the 2000s. This new type of bicycle-shared system authorizes the operator to identify the user and provides the ability to track its use. These innovations led to a significant reduction in bicycle robbery and vandalism. Programs like "Call a Bike" in Munich and "Velo'v" in France illustrated this generation, with steady growth throughout the decade. Their better-than-expected success changed the trajectory of bike-sharing history and created immense interest in this transit

mode from around the world. Starting in 2008, bike-sharing initiatives emerged in countries such as Brazil, China, and the USA (Albiński et al., 2018; Z. Chen et al., 2020; DeMaio, 2009; El-Assi et al., 2017; Eren & Uz, 2020; Fishman, 2016; Reiss & Bogenberger, 2015; Shaheen et al., 10/20212).

The development did not end at this point. The 4th generation of bike-sharing programs called “Multi-Modal Systems”, is identified by incorporating intelligent systems aimed at achieving sustainability, efficiency, and quality. Many modern public bike-sharing systems enable temporary bike rentals to go from one docking station to another. These docked systems are utilized with dynamic pricing schemes and are usually IT-based with credit card payments. The challenge of accessing docking stations is a barrier that prevents the incorporation of station-based bike-sharing services, and the number of docks is often constrained by space limitations in a city. Convenience in accessing a docking station is still one of the obstacles in bike-sharing systems (Shen et al., 2018). The most recent generation dockless bike-sharing systems have the capability to overcome this barrier. These new systems, called “dockless” or “free floating”, combine cashless mobile payments and GPS tracking and, ensure customers with enhanced flexibility, allowing bikes to be conveniently parked anywhere within the designated service area and eliminating the need for dedicated stations. The bikes can be parked at any suitable location within the operation area and the fleet size isn’t limited by the docking station (Albiński et al., 2018; Shen et al., 2018). Recently, hybrid approaches like Norisbike in Nuremberg or MVGRad in Munich have emerged. These systems enable users to both park and rent bikes either to docks or anywhere within the operational zone (Albiński et al., 2018).

*Figure 1. Docking Station of MVG Rad Munich (*MVG Rad*, 2023)*

Intelligent systems marked a new era in bike sharing, emphasizing advanced features and a holistic approach to address modern urban transportation challenges (Albiński et al., 2018; W. Chen et al., 2022; Z. Chen et al., 2020; Reiss & Bogenberger, 2015; Shaheen et al., 10/20212). As a result, over the past years, the attention to the idea of bike-sharing systems has extended worldwide (Bachand-Marleau et al., 2012; Z. Chen et al., 2020; Eren & Uz, 2020; Fishman et al., 2015; Liao et al., 2023; Shaheen et al., 2010). According to *The Meddin Bike-Sharing World Map Report* (2022), in August 2022, bike-sharing systems were present in 1590 cities and 92 countries. In the same report, it is also stated that there are 1914 schemes that have 8.967.122 bikes all around the world.

2.2 Bike-Sharing Systems & Users's Travel Behavior

The main factors influencing users' travel behavior within bike-sharing systems such as user demographics, land use & built environment, public transportation accessibility, station locations, and temporal aspects are heavily studied in the literature. This complex interaction is important in understanding the dynamics of bike-sharing systems and usage patterns.

Temporal Factors

The trip duration of shared bikes is uniform, ranging from 16 to 22 minutes. Bike-sharing trends display consistent patterns across various cities. Weekdays witness peak usage from 7 am - 9 am and 4 pm - 6 pm, serving primarily commuting purposes. However, on weekends, there's a shift with higher activity around midday, indicating leisurely rides (Z. Chen et al., 2020; Fishman et al., 2015). Moreover, bike-sharing systems also integrate with public transit, especially during the morning hours. Trips start from residential areas around 6:00 to 8:00 a.m. and transition to subway stations by 8:00 - 10:00 a.m., highlighting the role of bike-sharing as a "last mile solution" (Z. Chen et al., 2020). Weekends and public holidays show reduced bike activity, particularly during morning hours, while park visits increase significantly compared to weekdays. This points to a recreational trend during weekends (Eren & Uz, 2020). Beyond daily and weekly patterns, bike-share usage also shows seasonal differences in various cities. According to Z. Chen et al. (2020), in 2013, Barcelona maintained high year-round bike usage, while others like New York City and Paris experienced notable peaks both in September. In comparison, Washington, D.C. consistently achieves stable utilization, even during harsh winters, overtaking Australian cities like Melbourne and Brisbane. All in all, these findings emphasize the relevance of bike-sharing as a comprehensive transportation solution, meeting both commuting and leisure mobility needs across urban places.

User Demographics

There's a notable link between bike-sharing usage and demographic factors like gender, age, education, and income. Users of bike-sharing systems are typically young males with higher education, working professionals, and middle to higher incomes, primarily aged between 25 and 37 (Eren & Uz, 2020; Fishman, 2016; Fishman et al., 2015; Ji et al., 2020). Females mainly opt for shorter distances in the system, forming a smaller user group. Price levels significantly affect demand, especially in lower-income regions (Eren & Uz, 2020; Fishman, 2016). However, it's essential to recognize that these patterns are not universal, and variations can exist.

In their study, L. Zhang et al. (2015), classified the main users of bike-sharing programs within five Chinese cities. In Beijing, users are formed by commuters and tourists, Hangzhou is marked by mostly tourists and commuters, Wuhan's user base is mainly formed by commuters and shoppers, and Zhuzhou's bike-sharing users are predominantly city residents. It can be understood that although there are various groups of users in different cities, bike-sharing systems are mainly utilized by commuters.

As highlighted by Fishman (2016), bike-share user characteristics are influenced by various aspects such as higher income, education, and employment status. These characteristics can differ based on cultural backgrounds and geographic contexts. Gender participation in bike sharing varies across countries, with lower rates in regions with less cycling occurring, while countries with developed cycling cultures tend to exhibit more balanced gender participation. This emphasizes the role of cultural and geographical factors in structuring the demographic composition of bike-sharing program users.

While there are common patterns of user preferences, these preferences are shaped by cultural backgrounds and geographic contexts. Thus, a critical outcome should be evaluating each bike-sharing system must include its unique set of variables. The combination of various factors highlights the need for an adopted approach to understanding and harnessing the potential of bike-sharing programs within the complex structure of urban mobility.

Effects of Built Environment & Land Use

The influence of the built environment and land use on bike-sharing usage is extensively documented across various studies. In their study, Eren & Uz (2020), who evaluated factors that influence bike-share usage, stated that separated bike paths have been fundamental to establishing a safe and dependable environment for cyclists, encouraging bike usage, and attracting individuals who are not members. In the same study, they also mention that cyclists prefer paths & systems with devoted

infrastructure and enhanced safety illustrated with the lighting of the streets, secure parking, and rental options to increase access to the transit points at night time.

Population density, employment opportunities, and mixed land use exhibit positive correlations with bicycle usage. Commercial areas experience significantly higher demand on weekdays without rain compared to residential zones, while parks also show elevated demand. Proximity to amenities such as green spaces, educational institutions, commercial centers, and transit hubs positively influences bike-sharing program utilization (El-Assi et al., 2017; Eren & Uz, 2020; Ji et al., 2020; Lin et al., 2020; L. Zhang et al., 2015). The built environment factors such as encompassing density, land use diversity, bike lane length, and the distribution of points of interest, are shown to mold bike-sharing preferences (Lin et al., 2020).

Public transportation systems like subways and buses play a crucial role in driving bike-sharing usage. Demand is positively correlated with land use features including residential, office, and entertainment zones, while the impact of leisure and education areas is comparatively lower (Lin et al., 2020). Bike sharing can be a complementary mode of public transportation. In short distances, buses and bikes complement each other. In longer trips, highlight the substitution of buses for bikes. And bike-sharing systems emerge as an alternative solution particularly when public transport isn't available at night or in areas with weak transport options (Z. Chen et al., 2020; Eren & Uz, 2020). Bike-sharing systems also extend the reach of public transit, effectively serving as companions to walking transfers, which are usually shorter distances, while bike-based transfers cover slightly longer distances (Z. Chen et al., 2020).

The appeal of bike-sharing systems for first/last mile trips and the greater inclination of private bike owners to use these systems due to perceived safety advantages are highlighted by Fan et al. (2019). Access distances of 500 meters or less further boost the likelihood of travelers opting for bike-sharing over alternative transportation modes (Fishman, 2016). The importance of docking station proximity at home locations is a recurring theme across studies, reflecting its impact on promoting shared bicycle use (Bachand-Marleau et al., 2012). Station-level factors further shape bike-sharing demand. Station proximity to one another strongly influences usage, and factors like proximity to restaurants, museums, transit stops, schools, sports facilities, and shopping centers play significant roles. The availability of stations around users substantially boosts demand, particularly in affluent neighborhoods (Eren & Uz, 2020).

Shen et al. (2018) studied land use density across categories such as public residential, private residential, commercial, and industrial areas. They found that public residential density exhibited a negative correlation with bike usage, potentially due to an oversupply of bikes. In contrast, private residential density had a small positive effect, and higher commercial land use density was linked with increased usage, possibly due to last-mile trips. Land use diversity demonstrated a positive

correlation with bike usage. Cycling facilities and transportation infrastructure were found to significantly encourage dockless bike usage (Shen et al., 2018).

In conclusion, interactions between the built environment, land use, transportation infrastructure, and station attributes have an important impact on bike-sharing usage patterns. The emphasis on safe infrastructure, integration with public transit, strategic station placement, and proximity to key destinations collectively contribute to promoting sustainable transportation choices and enhancing urban mobility.

2.3 Spatiotemporal Analysis of Bike-Sharing Systems

In their study, Y. Zhang et al. (2017) employed a multiple linear regression model to investigate the influence of built environment factors on travel demand and the demand-to-supply ratio (D/S) at bike stations. They combined station attributes, accessibility, cycling infrastructure, public transport facilities, and land use characteristics as major factors in their analysis. They also assessed the spatial correlations of bike share usage between nearby stations using a spatial weighted matrix. The researchers have found that population density, bike lane length, secondary road length, mixed land-use types, and the number of nearby stations positively affected both trip demand and D/S, while the range to the city center had a negative influence.

In the study by Rixey (2013), where the author investigated station-level effects in three different U.S. cities' bike-sharing systems, multivariate linear regression and network effects analyses were conducted to reveal the connections between bike-sharing ridership and various factors. The findings displayed that a range of variables, including total population, job types, income levels, alternative commuting habits, education, parks, bikeways, and more, significantly influenced bike-sharing ridership. The study also evaluated the impact of the bike-sharing station network, finding that network effects played a crucial role in shaping ridership patterns across the analyzed systems.

Albiński et al. (2018) analyzed Munich's hybrid bike-sharing performance under censored demand. In their study, the operation area is partitioned into hexagons, and a data-driven approach for estimating bike-sharing demand is implemented. Two service level metrics are identified to measure the performance of the system: the α-service level, which measures the availability of the system, and the β-service level, which measures the fill rate or the portion of requests that can be satisfied in a zone. They have found that reservations are mainly made for free-floating bikes which indicates that customers prefer to use bikes close to their location rather than going to stations. Moreover, the average α-service levels and β-service levels across the operating area reveal the differences between highly-used districts and low-used districts. The availability and fill rates are generally higher in highly used districts and

at bike stations and the system performs better in highly utilized areas in comparison to low-utilized regions.

In another study, Reiss & Bogenberger, (2015) studied Munich's first public bike-sharing system, "Call a Bike" of German Railways, to reveal insights by employing Temporal and Spatial Analysis Methods. The study analyzes the number of trips per day, the spatial distribution of rentals and returns, and the frequency of rentals per user to reveal trends, spatial patterns, and user behavior. It is found that heavy users which is approximately %20 of all customers, account for %80 of all trips, and more than %50 of customers ride a bike less than 5 times a year. Regarding temporal aspects of the system, the majority of the trips occur during the summer months due to good weather conditions. Weekdays show peaks during morning and evening hours due to commuters while weekends exhibit smoother activity patterns. Furthermore, it is also found that stationless bikes show more complex spatial patterns than docked ones, and bikes tend to move from residential areas to the city center in the mornings and vice versa in the evenings.

Zhou (2015), analyzed spatiotemporal patterns of the bike share system in Chicago, and bike flow patterns are revealed by identifying neighboring flows for each trip and similar flows grouped into clusters with the community detection algorithm. To study the spatiotemporal demand of the system, hierarchical clustering methods were used to cluster stations with similar patterns. The author found that the bike flow patterns varied based on time, weekdays/weekends, and user types. Inbound trips dominated during morning peak hours, with many trips traveling into downtown areas. Outbound trends were observed during afternoon peak hours. Different clusters of trips were identified, indicating distinct travel patterns. For example, on weekends, customers show clustered trips in recreational areas.

McBain & Caulfield (2018), analyzed influencing factors of trip duration variation in the public bike-sharing system of Cork, Ireland. The researchers applied Multinomial Logistic Regression to analyze journey data and identify patterns. Journey time variation accounted as the dependent variable and multiple independent variables such as spatial and temporal variables are used. The results indicated that the busiest bike-sharing stations were located on the city outskirts and near major destinations. Temporal patterns showed morning and evening peaks, along with substantial inter-peak and weekend usage. The regression models stressed that variables like one-way streets, station types, cycle-friendly routes, nearby amenities, public transport links, and user type as influential factors in trip duration variations.

2.4. Community Detection

2.4.1. Basic Concepts

The theory of graphs evolved with Euler's Königsberg bridges puzzle solution in 1736 (Euler, 1736, 1953). Over time, a lot of progress has been made in understanding mathematical properties, and, in the 20th century, graphs found application in diverse domains like biology, social sciences, engineering, technology, and information networks (Fortunato, 2010). Graphs were utilized as representations for real-world systems, and analysis methods became crucial. Recently, computer-related innovations allowed scientists to deal with big data, and computational resources to analyze these data. This shift prompted new approaches to studying graphs (Danon et al., 2005; Fortunato, 2010; Fortunato & Hric, 2016).

A community is a group or cluster of elements where elements inside are closer to the other elements of that community than the elements outside of it. Consequently, in an interacting event that is depicted, groups of elements that supposedly share prevalent features and/or play similar roles are called communities. Community detection is crucial in many aspects such as the classification of nodes in a network which causes uniform clusters, group leaders, or crucial group connectors. For example, they can resemble World Wide Web pages with related topics, groups of connected people in social networks, and so on (Coscia et al., 2011; Danon et al., 2005; Fortunato & Hric, 2016).

Community identification can reveal underlying patterns and hidden structures in a network. It enables us to concentrate on parts of the graph holding a certain level of autonomy. Moreover, it helps to categorize the vertices of the network according to their role within the community(Fortunato & Hric, 2016). To illustrate, it is possible to differentiate vertices that are deeply integrated within their cluster and vertices that are on the boundary. The boundary vertices might have an important role due to defining both holding the modules together and information spreading across the network (Fortunato & Hric, 2016).

2.4.2 Fundamentals of Graph Theory

The theory of graphs evolved with Euler's Königsberg bridges (Figure 2) puzzle solution in 1736 (Euler, 1736, 1953; Fortunato, 2010; Gribkovskaia et al., 2007). In the 18th century, there was a debate among citizens of the city Königsberg, now called Kaliningrad, Russia, if it was possible to have a closed walk by visiting seven bridges of the Pregel River only once and returning to the starting point. Swiss Mathematician Leonhard Euler (1707 – 1783), was chair of mathematics at the St. Petersburg Academy of Sciences, and solved the problem (Euler, 1736, 1953; Gribkovskaia et al., 2007). The city is divided by the River Pregel into four pieces and these landmasses are connected by seven bridges. To be able to solve the problem, Euler, abstracted

the geography by utilizing a graph, which has nodes referring to landmasses, and bridges as edges (Euler, 1736, 1953; Gribkovskaia et al., 2007). He found that to have a closed walk, each landmass in the city must have even numbers of bridges. Since four landmasses have an odd number of bridges, it is proven that it is not possible to have a closed walk. Such closed path is now called unicursal or Eulerian Path. The Bridges of Königsberg is one of the most well known problem in mathematics, a foundation for graph theory and is a classic practice for topology (Euler, 1736, 1953; Gribkovskaia et al., 2007).

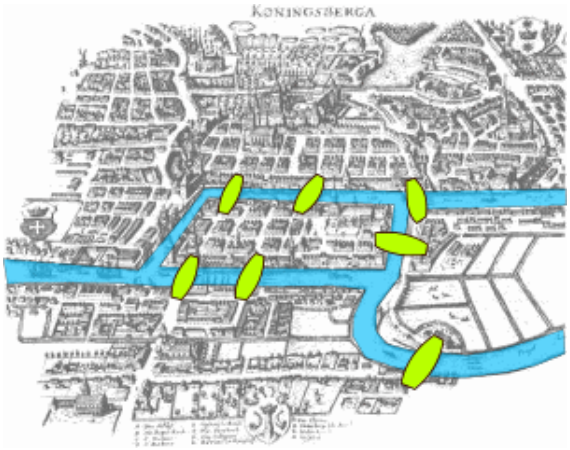


Figure 2. Illustration of Seven Bridges of Königsberg (Giuşcă, 2011)

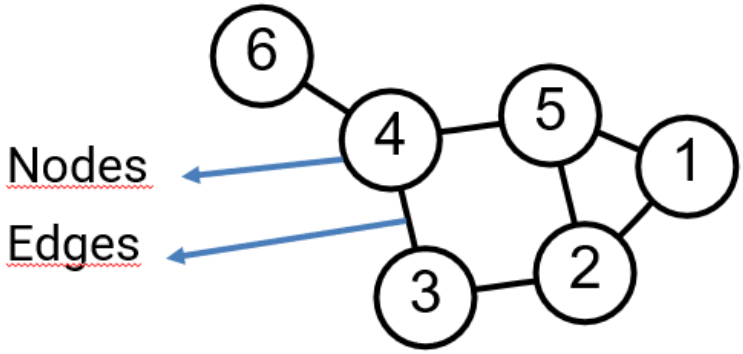


Figure 3. Simple Graph Representation

Graphs can be defined as a set of objects and relations between pairs of objects. A graph G is a pair of two sets (V, E), where V is the group of vertices or nodes, and E, a subset of V^2, is the group of disorganized pairs of elements of V. Elements of E connects vertices, named edges or links, and vertices at the two ends of an edge called endpoints. An edge is always adjacent to each of its vertices. If edges are ordered pairs of vertices, then it is a directed graph where (v, w) shows an edge from v to w. Graphs can be visualized as points connected by lines (Figure 3). In their real-world examples, numbers can be assigned to their edges and thus, graphs are

weighted. Moreover, graphs do not include self-loops where an edge starts and ends at the same vertex (Dao et al., 2021; Fortunato, 2010; Kulikov, n.d.).

The complete knowledge regarding the topology of a graph with N nodes encompassed in the adjacency matrix A, where A is a N x N matrix, and component Aij is equivalent to 1 if vertices i and j are joined by an edge, if not, it is zero. Because there are no loops present, the diagonal entries of the adjacency matrix are all assigned zero (Barthélemy, 2011; Fortunato, 2010; Phillips et al., 2015). If A belongs to an undirected graph, A is a symmetric matrix, and Aij is equal to Aji. If the graph is unweighted, components of the adjacency matrix can only be equivalent to either 0 or 1. If the graph is weighted, one can derive the weight matrix W, whose component Wij could have different values than 0 or 1 as the weight (Fortunato, 2010; Phillips et al., 2015).

2.4.3 Community Detection and Network Representation

Many real-world networks are very broad therefore, one must facilitate their structure before useful information can be derived about the systems they represent (Rosvall et al., 2009). The initial problem in graph clustering is to search for a quantitative description of the community. There is no description that is globally accepted. The definition usually relies on the certain system at hand and/or application. In other words, communities are algorithmically defined, the final result of an algorithm, and without a certain theoretical definition (Fortunato, 2010). The main goal of community detection is to partition vertices of a network into several k groups, where the number of edges within these groups is maximized and the number of edges between vertices of groups are minimized (Coscia et al., 2011; Danon et al., 2005; Fortunato, 2010; Song et al., 2021).

In order to facilitate analysis within networks, a definition of basic concepts of community detection is necessary. If one considers a subgraph C of a graph G, with $|C| = nc$ and $|G| = n$ represents the total number of vertices. Each vertex v in C, $v \in C$, has an internal degree k_{int}^{v}, as the number of connections within C, and external degree k_{ext}^{v}, representing connections to the rest of the graph G. In this case, if the external degree of a vertex v equals zero, $k_{ext}^{v} = 0$, the vertex has neighbors only within the subgraph C, which indicates a good cluster for v. On the other hand, if $k_{int}^{v} = 0$, the vertex is not in C, and would be better to assign it to a different cluster. The sums of internal and external degrees in C are $\mathrm{k}_{\mathrm{int}}^{\mathrm{C}}$ and k_{ext}^{C} respectively. The total degree is $k^{C} = k_{int}^{C} + k_{ext}^{C}$ (Fortunato, 2010). The intra-cluster density $\delta_{int}(C)$ of the subgraph C could be defined as ratio of the internal edges of C and the number of all possible internal edges. Similarly, the inter-cluster density $\delta_{ext}(C)$ is the ratio between vertices connecting outside of the C and the maximum possible inter-cluster edges (Fortunato, 2010). Two notations can be formulized as;

$$\delta_{int}(C) = \frac{number\ of\ \text{internal}\ edges\ in\ C}{\frac{\text{nc}(nc-1)}{2}}$$

Equation 1 Intra Cluster Density of Subgraph C (Fortunato, 2010)

$$\delta_{ext}(C) = \frac{number\ of\ \text{inter} - \text{cluster}\ edges\ from\ C}{nc(n-nc)}$$

Equation 2 Inter Cluster Density of Subgraph C (Fortunato, 2010)

If C is considered as a community, it is anticipated that $\delta_{int}(C)$ prominently surpass mean link density $\delta(G)$ of the graph G. $\delta(G)$ can be defined as the ratio between the actual number of edges in G and the number of maximum possible edges within G (Fortunato, 2010). Conversely, $\delta_{ext}(C)$ should be much smaller than $\delta(G)$. Many community detection algorithms try to find stability between high $\delta_{int}(C)$ and low $\delta_{ext}(C)$, and one of the main approaches is maximizing the difference $\delta_{int}(C) - \delta_{ext}(C)$ across clusters (Fortunato, 2010).

The process of community detection is instructed by a quality function, that helps to determine how well a distinct clustering captures the underlying structure of the network. The aim is to observe meaningful clusters and consistent algorithms are needed for this procedure. Describing what forms a good clustering is fundamental. For this purpose, specific attributes are required, that are widely agreeable (Fortunato, 2010; Fortunato & Hric, 2016). Various algorithms exist that can locate valuable clustering, some only few, and others deliver a large number of clusters. However, not all partitions are equally good. Therefore, a quantitative criterion is necessary that is represented in a quality function to evaluate the effectiveness of community detection (Fortunato, 2010; Fortunato & Hric, 2016). The quality function assigns a numerical value to each clustering, allowing for the ranking of partitions based on their scores. Ultimately, the clustering with the highest score is considered the best representation of the partition (Fortunato, 2010; Fortunato & Hric, 2016).

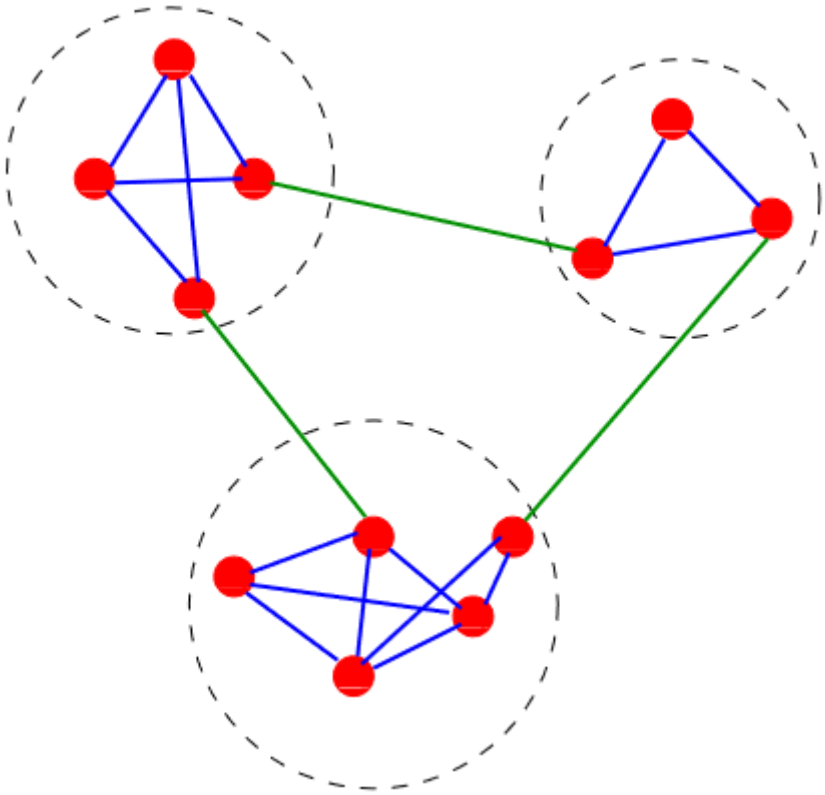


Figure 4.. A simple graph representation with three communities (Fortunato, 2010).

Social networks have been studied widely for decades, representing relationships among individuals. These networks can be broad, such as scientific collaboration, mobile calls, and online interaction, offering valuable insight. Communities in these networks can be friend circles, interest groups, etc. Social networks symbolize individuals as nodes and interactions as edges. To illustrate, Blondel et al. (2008), analyzed mobile call networks and revealed linguistic divisions. Tyler et al. (2003), studied email exchanges that match company departments. Traud et al. (2011) used Facebook data where authors linked students based on friendships. Social networks, thus, offer insights into human connections, visualized as graphs with nodes and edges (Fortunato, 2010).

The availability of broad data allows quantitative research and modeling of spatial networks. The main examples of spatial networks are transportation, infrastructure, and mobility networks (Barthélemy, 2011). Transportation systems can be abstracted as graphs where nodes display certain features such as airports, stations, administrative units (neighborhood, municipality, city, country, or continent), or ports. Links (edges) between nodes indicate flows or connections between these entities. The edges between nodes can be directed and weighted as they can show direction and the number of flows in between entities (Barthélemy, 2011). Thus, the network representation of transportation systems provides an understanding of the structure, connectivity, and spatial relationships. Infrastructure networks can be mapped as

graphs, where nodes symbolize key elements such as intersections, power plants, routers, or distribution hubs (Barthélemy, 2011). The connections or flows between these nodes are presented as links in the graph. For road and street networks, links represent roads, while for power grids and water distribution networks, links indicate transmission lines or distribution pathways. In the case of the Internet, routers are nodes connected by links. This graph-based representation provides a brief way to analyze the spatial structures, topological attributes, and interactions within these crucial infrastructure systems (Barthélemy, 2011).

Human mobility data is crucial for understanding spatial economics, disease transmission, and optimizing business strategies, and helps to optimize mobility systems based on people's movement patterns. To reveal human travel behavior, a common approach is to divide the study area into zones labeled as i = 1, ..., N and measure the number of individuals moving between zones i to j, forming an origin-destination (OD) matrix (Barthélemy, 2011). This matrix adopts the logic of an adjacency matrix and supports transportation models and network analysis. Technological advancements such as GPS, mobile phones, and geosocial apps have allowed more accurate measurements on broad datasets and offer a better understanding of urban movement patterns (Barthélemy, 2011).

$$\begin{bmatrix} A_{11} & \cdots & A_{1j} \\ \vdots & \ddots & \vdots \\ A_{i1} & \cdots & A_{ij} \end{bmatrix}$$

Figure 5. A simple example of an Adjacency Matrix

2.5. Community Detection & Bike – Sharing Systems

Community Detection in bike-sharing systems is crucial for gaining insights into travel patterns, operation optimization, and decision-making in urban planning. The technique includes building spatial networks by using users' bike trip data and implementing complex algorithms to identify clusters or communities of closely connected traffic zones. These clusters reflect unique travel behaviors, demand distribution, and interaction patterns within the system. Bike-share operators and urban planners can customize their plans on specific areas, times, and user groups to improve the efficiency, resource allocation, and infrastructure of the systems (W. Chen et al., 2022; Lin et al., 2020; Lu et al., 2023).

Song et al. (2021), implemented a spatiotemporal dynamic analyses approach that focuses on analyzing cycling activities in a dockless bike-share system, aiming to enhance decision-making for system operation and transportation planning. The

research comprises three main components: data preprocessing, spatial traffic distribution modeling, and analysis of traffic spatiotemporal variations. They examine cycling patterns in Singapore's bike-share system using an eight-day dataset divided into peak and non-peak hours. For community detection, they employ the Louvain algorithm which has two iterative phases. The algorithm first assigns each node to a community. Next, node i is removed from its community and assigned to neighboring nodes's communities based on the modularity calculations. The results show stable spatiotemporal origin-destination (OD) demands, clustered cycling trip distributions, and robust community structures. The top 15 largest communities' scale varies significantly between weekdays and weekends, indicating temporal variations. Spatial communities remain stable throughout the day, with weekday' communities having more consistent shapes than weekends'.

W. Chen et al. (2022), studied delineating urban activity zones using free-floating bike sharing (FFBS) journey data from Nanjing, China, to better understand travel patterns and urban spatial structure. They employ the Leiden algorithm as the community detection method to build an FFBS spatial interaction network and identify activity zones. The results indicate the presence of 22 activity zones, with tight connections within each zone and looser connections between zones. This division of activity zones provides a more rational separation of FFBS travel flows compared to traditional administrative districts. The study highlights the potential for improved bike rebalancing algorithms, better evaluation of urban policies, and an understanding of the influence of different types of activity zone borders on travel flows.

Lu et al. (2023), recently studied the spatiotemporal characteristics of Dockless Bike-Share trips and proposed a research framework for large-scale bike rebalancing at the city level. The framework integrates tools like GPS data visualization, management-area detection, virtual-station identification, and rebalancing-scheme generation. The research uses 1-week bike-sharing data from Shanghai and explores phenomena and conclusions regarding bike-sharing travel behavior and rebalancing strategies. Key findings include higher travel volume and riding speed during peak weekdays, division of Shanghai into 28 management sub-areas based on geographic features, identification of 1,190 virtual stations in the central urban area, and the efficiency of management-area-based rebalancing compared to administrative divisions. The proposed framework aids policy implementation, system management improvements, and insights for bike-sharing companies.

Lin et al., (2020), analyzed the temporal and spatial patterns of dockless bike-sharing trip demand using various data sources, including Mobike trip data, POI data, and smart card data. They employ the Infomap algorithm as a community detection method to understand the spatiotemporal usage patterns of bike sharing. Furthermore, they utilize Gradient Boosting Decision Tree (GBDT), a machine learning technique, to uncover the factors influencing bike sharing demand, considering

aspects like the built environment, public transit ridership, and temporal factors. The results revealed the imbalanced distribution of bike-sharing trips, with hotspots around core areas within the city. Community detection reveals a polycentric pattern of trip demand distribution with self-contained subregions. The GBDT model uncovers the impact of factors on bike sharing demand, with public transit ridership being a significant contributor, along with built environment variables. The study provides insights for operators and planners to optimize bike redistribution, parking locations, and overall system efficiency.

Zhou (2015) focused on analyzing the spatiotemporal patterns of bike-sharing behavior in Chicago using massive bike-sharing system data from July to December 2013 and 2014. They use the fast-greedy algorithm to detect unique travel patterns on weekdays and weekends, as well Lin et al., (2020), analyzed the temporal and spatial patterns of dockless bike-sharing trip demand using various data sources, including Mobike trip data, POI data, and smart card data. They employ the Infomap algorithm as a community detection method to understand the spatiotemporal usage patterns of bike sharing. Furthermore, they utilize Gradient Boosting Decision Tree (GBDT), a machine learning technique, to uncover the factors influencing bike sharing demand, considering aspects like the built environment, public transit ridership, and temporal factors. The results revealed the imbalanced distribution of bike-sharing trips, with hotspots around core areas within the city. Community detection reveals a polycentric pattern of trip demand distribution with self-contained subregions. The GBDT model uncovers the impact of factors on bike sharing demand, with public transit ridership being a significant contributor, along with built environment variables. The study provides insights for operators and planners to optimize bike redistribution, parking locations, and overall system efficiency.

3. Methodology

The methodology section presents a systematic approach for the detection and analysis of bike-sharing communities within mobility networks. It involves several key stages, beginning with data preprocessing, where relevant parameters are selected and data is cleaned to ensure consistency and reliability. The subsequent steps encompass static community detection, employing the InfoMap algorithm to reveal communities at different time snapshots, thereby shedding light on temporal patterns. Additionally, dynamic community detection is introduced, leveraging consensus clustering to identify evolving communities throughout the day. Afterward, a dynamic sub-community detection process that applies the InfoMap algorithm to the result of consensus clustering is introduced. Finally, the methodology incorporates quality criteria, with a focus on modularity as a quantitative measure to evaluate the effectiveness of community detection. Through this comprehensive process, the study aims to uncover and analyze bike-sharing communities, offering valuable insights into the intricate patterns of human mobility.

3.1. Data Preprocessing

In this section, we provide an overview of the data preprocessing methodology undertaken to prepare the collected datasets for the subsequent analysis of bike-sharing communities. The preprocessing steps are designed to enhance data quality and consistency across various data sources. The raw bike-sharing mobility data typically includes various parameters and non-structuralized records. Before processing the data, data selection and cleaning are important steps

Parameter Selection

The selection of relevant parameters from the collected datasets is a crucial initial step in the data preprocessing process. The chosen parameters serve as the foundation for community detection analysis. The proposed framework can be adopted to any type of mobility data as long as the following parameters are selected:

- Location Information: Origin-Destination (OD) data, which encompasses the spatial locations of mobility trips.
- Time Record: Date and time information associated with each mobility transaction.
- Spatial Units: The spatial granularity at which we analyze the data.

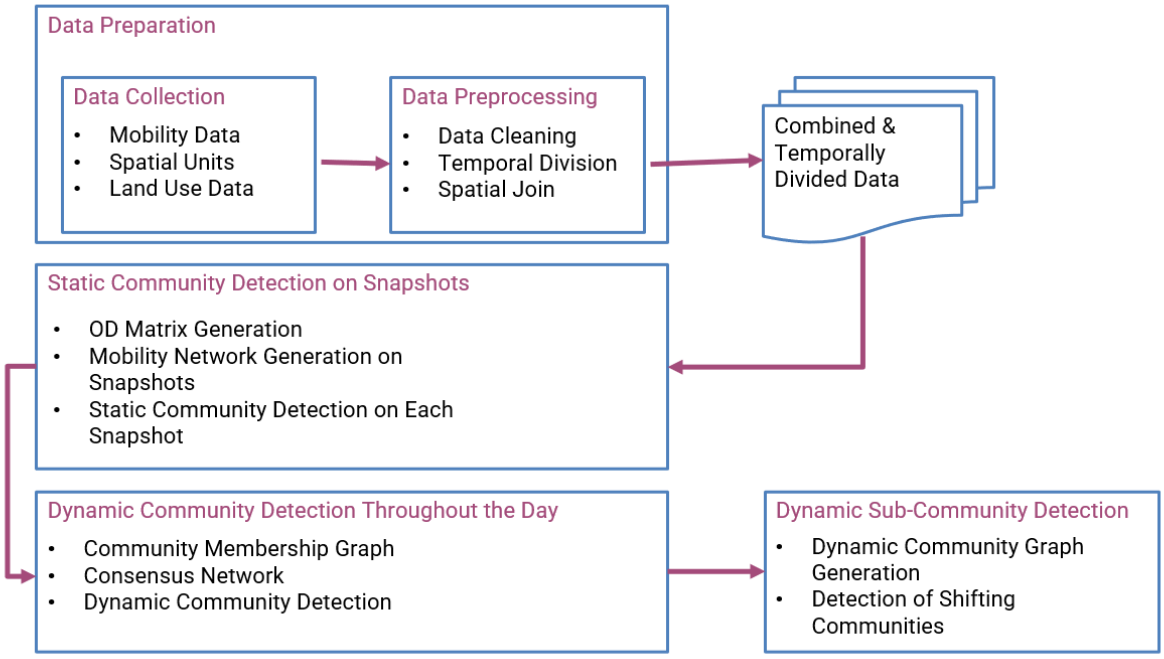


Figure 6. Workflow of the proposed framework

Data Cleaning

Data cleaning is a crucial step because raw data is frequently flawed with missing entries, discrepancies, and outliers. Ensuring data consistency and accuracy is vital, so we clean each data source in the following manner:

Table 1 Community Detection with Mobility Data Parameter Selections

Field Name	Field Value	Description
Start Time/End Time	1/1/2022 0:36	string
Origin/Destination Latitude	48.15791	Double float
Origin/Destination Longitude	11.52835	Double float
Spatial Unit	1, …, N (N = Total Number of Districts)	Object ID

Mobility Data:

- Cleaning GPS Errors: Identification and removal of trips with origin and destination points falling outside the predefined study area, eliminating GPS errors.
- Deleting Missing Values: Removal of data points with missing or incomplete information.
- Eliminating Duplicate Trips: Removal of trips with identical origin and destination locations, which may not contribute meaningful insights and could be indicative of bike malfunctioning.

Spatial Units:

- Combination of Spatial Levels: The study area should be partitioned into spatial units according to intended spatial resolution. Either with spatial grids or with pre-defined authoritative administrative districts, one should generate or select units only within the study area.

Land Use Information:

- Data Selection: Careful selection of relevant land use information to incorporate into our analysis. Selected land use information will be assigned to spatial units.

3.2. Static Community Detection on Snapshots

The first part of the proposed method is to detect static communities at snapshots. We partition 24 hours of a day into $\tau = 24/\Delta$ snapshots using a pre-given interval, e.g., $\Delta = 2$ hours. Accordingly, the dynamic OD network $G = \{G^1, \dots, G^t, \dots, G^{t+x}\}$ are constructed. Each snapshot G^t is represented as a directed and weighted graph, and each weight represents traffic flows between the OD pair during t^{th} time interval. The well-developed static community detection method, InfoMap, is employed in this study for detecting communities $C^t = \{C_1^t, \dots, C_i^t, \dots, C_k^t\}$ at each snapshot G^t.

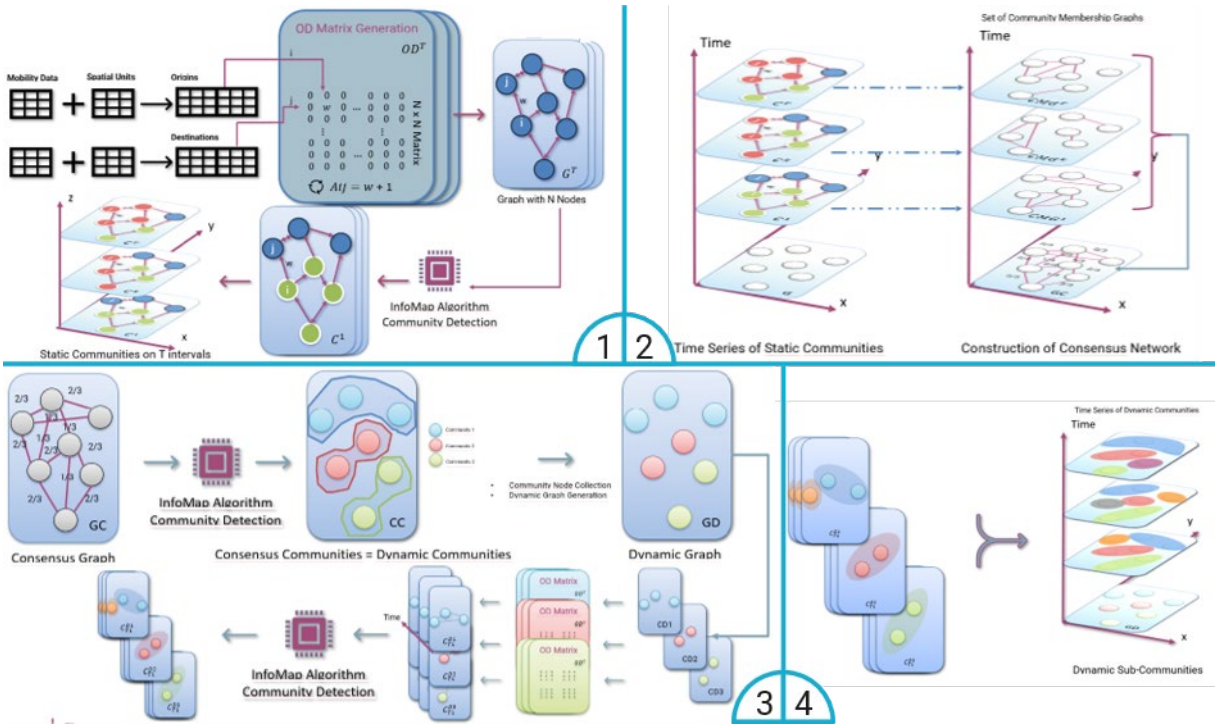


Figure 7. Graphic abstract of the proposed framework.

Division of Mobility Trips

The entire year's worth of bike-sharing data is divided into time intervals, enabling the analysis of temporal patterns and variations in community detection. The 2-hour time interval is defined as an iterative process during the development of the proposed workflow. We have tried different intervals such as 30 minutes, however, in static community detection, the best results were acquired via 2-hour intervals. (***Please note*** that, the time unit we have decided while detecting static communities is different than what we applied in the dynamic sub-community detection part.)

Spatial Join & District Assignment

A spatial join operation is employed to determine which spatial unit (neighborhood or administrative) a given origin or destination point belongs to. This operation facilitates the spatial analysis of bike-sharing communities based on location data.

Land use information is assigned to specific spatial districts based on their respective area coverage within each spatial unit. This assignment helps correlate land use characteristics with bike-sharing activity within specific regions.

OD Matrix Generation

To be able to detect communities in our bicycle network, first, we need to create a graph that consists of nodes and edges. At this point, as we discussed, nodes are our

spatial units with no weights, and, edges are traffic flows between districts with weights of the number of trips. To assign these bicycle trips as weights to our edges, first, we need to define possible trip directions between districts (from their index) and then we need to count how many trips occurred between each district pair. This is achieved with adjacency matrix A, where A(i,j) records the weight of the edge from node I to node j. The size of the matrix is defined with the square of the district count.

Graph Generation

After having the adjacency matrix, a graph that consists of nodes and edges can be created. This can be achieved by using functions of the networkx library. A directed graph is created by passing the dataframe version of the adjacency matrix as input. A graph is automatically created by iterating over each OD pair and assigning counts of bicycle trips to each possible edge.

Community Detection with InfoMap Algorithm

There are various algorithms developed for the detection of communities within the network. The selection of an appropriate community detection algorithm is based on specific applications or intended results. Moreover, community detection within networks is a computationally difficult task therefore, one should utilize the most reliable and efficient algorithm to find meaningful structures within a network (Fortunato, 2010; Fortunato & Hric, 2016). In our study, the InfoMap algorithm (Rosvall et al., 2009; Rosvall & Bergstrom, 2008) is utilized to detect community structures. The algorithm was selected for the proposed framework due to showcasing the best trade-off between accuracy and computational performance based on benchmark tests that are conducted in several studies (W. Chen et al., 2022; Fortunato, 2010; Fortunato & Hric, 2016).

In their study, Rosvall and Bergstrom (2008) introduced the Map Equation, a fundamental concept in community detection within complex networks. They aimed to find the most efficient way to describe an infinitely long random walk on a graph, where information content is quantified by the number of bits required. Initially, a simple description involves listing all vertices reached by the random walker, each with a unique codeword. However, if the network exhibits a community structure, a more concise description can be achieved, similar to how geographic maps reuse street names across regions. In this context, communities act as regions, and vertices with the same name are distinguished by specifying their community (Lancichinetti & Fortunato, 2012).

The Map Equation quantifies the description length of an infinite random walk, consisting of two terms representing the Shannon entropy within and between clusters (Figure 6). The objective is to find a partition that minimizes this description length, a method known as Infomap. Infomap can be applied to weighted networks,

both undirected and directed. For directed networks, a teleportation probability is introduced to ensure a nontrivial stationary state. Infomap has also been extended to detect hierarchical community structures and overlapping clusters (Lancichinetti & Fortunato, 2012; Rosvall et al., 2009; Rosvall & Bergstrom, 2008).

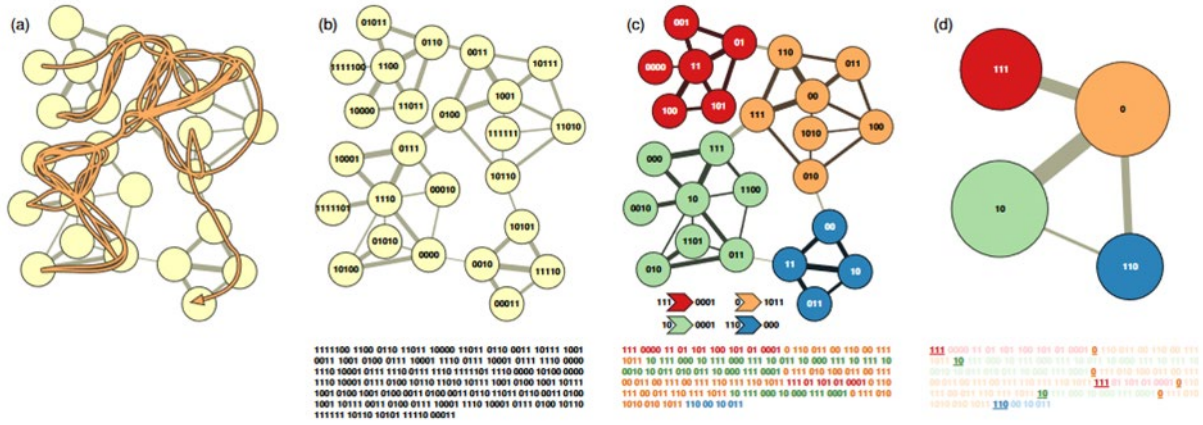


Figure 8. Maps of Random Walks.

Note. A method for efficiently encoding the trajectory of a random walker on a network using a two-level description approach. The first level uses codewords from a Huffman codebook, while the second level involves switching between module codebooks to reduce the description length. Retrieved from "The Map Equation" by Rosvall et al., 2009), The European Physical Journal Special Topics, 178(1), 13–23. https://doi.org/10.1140/epjst/e2010-01179-1

3.3. Dynamic Community Detection Throughout the Day

The second stage of the proposed method is to develop a consensus clustering method to detect the consensus structures as the dynamic communities of human mobility networks throughout the day.

Community Membership Graphs

This stage aims to build a consensus network using the snapshots of static communities, $\{C^1, \dots, C^t, \dots, C^k\}$, detected in the first stage. For every identified community $C_i^t \in C^t$, a community membership graph is generated by attaching an undirected edge between any two nodes within the same community. This community membership graph is undirected and unweighted (Zhao et al., 2023). In our methodology, there need to be 12 community membership graphs generated for 12 snapshots of 2-hour intervals.

Consensus Network

Subsequent to community membership graphs for all communities at all snapshots are constructed, they are reflected in the geographical space to produce a consensus network, represented as G^C. The consensus network $G^C = (V^c, E^c)$ is an undirected but weighted graph, where $V^c = \cup_{\forall i, \forall t} C_i^t$ is the union of all nodes within all communities $\forall C_i^t \in C^t$ at all snapshots $\forall C^t$; and E^c is the unions of all edges in all community membership graphs at all snapshots (Zhao et al., 2023). The weighting of an edge $e_{ij}^c \in E^c$, denoted by w_{ij}^c , can be expressed as

$$w_{ij}^c = \frac{\sum_{\forall c^t} \sum_{\forall c_i^t \in c^t} \delta\left(v_i^t, v_j^t, c_i^t\right)}{\tau}$$

Equation 3. Weight of an Edge in Consensus Graph (Zhao et al., 2023)

where $\delta\left(v_i^t, v_j^t\right)$ is the binary index; $\delta\left(v_i^t, v_j^t, c_i^t\right) = 1$ means that both nodes v_i^t and v_j^t within the community C_i^t; and $\delta\left(v_i^t, v_j^t, c_i^t\right) = 0$ otherwise. This w_{ij}^c represents the time duration of edge e_{ij}^c holding community membership and therefore quantifies the strong interaction between two nodes from the temporal perspective. This $\mathrm{w}_{\mathrm{ij}}^{\mathrm{c}}$ is standardized by the total number of snapshots, i.e., τ (Zhao et al., 2023).

Dynamic Community Detection

The following step is to identify consensus structures as dynamic communities of human mobility networks throughout the day. These consensus structures are detected by applying the InfoMap algorithm on consensus network G^C that are obtained in the prior step. Accordingly, a set of consensus communities, denoted by $C^C = \{\mathrm{C}_1^{\mathrm{C}}, \ldots, \mathrm{C}_{\mathrm{t}}^{\mathrm{C}}, \ldots, \mathrm{C}_{\mathrm{k}}^{\mathrm{C}}\}$, is determined on G^C (Zhao et al., 2023).

3.4. Dynamic Sub-Community Detection

The next step is to detect sub-communities within each C_i^D and quantify the time series of evolving events of sub-communities. Within a snapshot $C_i^{D,t}$, we can establish a equivalent sub-graph of $G^t = (V^t, E^t)$ denoted by as $G_i^{D,t} = (V^{D,t}, E^{D,t})$ where $V^{D,t} = C_i^{D,t}$, sub-graph holds same vertices of the original graph while $E^{D,t}$ contains all edges $e^{D,t} \in E^t$ whose origin $v_i^{D,t}$ and destination $v_j^{D,t}$ belong to $C_i^{D,t}$ (Zhao et al., 2023).

$w_{ij}^{D,t}$ is the number of journeys between OD pair during the t^{th} time interval. Specifically, this step again employs the InfoMap algorithm to detect a set of sub-communities, $SC_i^{D,t} = \{sc_{i,1}^{D,t}, \ldots, sc_{i,p}^{D,t}, \ldots, sc_{i,k}^{D,t}\}$ on the above $G_i^{D,t}$ (Zhao et al., 2023).

Dynamic Graph Generation

Since spatially & and temporally connected consensus communities identified, now we can find community shifts (sub-communities) within each consensus community. To be able to find sub-communities we need to generate separate graphs for each community and for each snapshot. In this part, we partition 24 hours of a day into $\tau = 24/\Delta$ snapshots using a pre-given interval, e.g., $\Delta = 0.5$ hours. We have chosen 30-minute time intervals to considering the average usage times of shared bicycles that are stated in the literature and to capture significant changes in human mobility patterns throughout the day (Eren & Uz, 2020; Zhao et al., 2023). Therefore if we have 6 consensus communities and have 48 time intervals than we will have 6 x 48 = 288 graphs at the end. The nodes of the dynamic graphs are identical to nodes of the corresponding consensus community, however, to be able to find community shift, edges are assigned from actual trips for each snapshot.

Sub-Community Detection

In the last step, once again infomap algorithm is applied to each dynamic graphs $G_i^{D,t}$ separately to detect community shifts (sub-communities) throughout the day. The result of the infomap algorithm reveals connected regions within a certain period in the study area. Subsequently, we can finally combine detected sub-communities $SC_i^{D,t} = \{sc_{i,1}^{D,t}, \dots, sc_{i,p}^{D,t}, \dots, sc_{i,k}^{D,t}\}$ within t^{th} time interval.

3.5. Quality Criteria

Modularity

The process of community detection is instructed by a quality function, that helps to determine how well a distinct clustering captures the underlying structure of the network. The aim is to observe meaningful clusters and consistent algorithms are needed for this procedure. Describing what forms a good clustering is fundamental. Therefore, a quantitative criterion is necessary that is represented in a quality function to evaluate the effectiveness of community detection. The quality function assigns a numerical value to each clustering, allowing for the ranking of partitions based on their scores. Ultimately, the clustering with the highest score is considered the best representation of the partition (Fortunato, 2010; Fortunato & Hric, 2016).

Modularity, a quality function, evaluates both the internal intensity of a community and the absence of edges among communities (Coscia et al., 2011). The primary quality function is the modularity by Newman and Girvan (Newman & Girvan, 2004). It approximates the quality of a partition of the network in communities (Fortunato & Hric, 2016). According to Coscia et al. (2011), Newman's modularity maximization strategy repeatedly merges the two communities whose join produces the utmost increase in quality function Q. The general expression of the modularity is:

$$Q = \frac{1}{2m}\sum_{ij}(A_{ij} - P_{ij})\delta(C_i, C_j),$$

Equation 4. Modularity Function (Coscia et al., 2011)

where m is the number of edges of the network, the sum iterates over all vertice pairs of i and j, A_{ij} is the component of the adjacency matrix, P_{ij} is the null model notation, and the delta δ at the end C_i and C_j indicate communities i and j of the network. If two nodes are in the same community $(C_i = C_j)$, $\delta(C_i, C_j) = 1$, if not the delta is zero. The P_{ij} corresponds to the mean adjacency matrix of the combination of networks which is derived by randomising the original graph while preserving some of its attributes. Thus, the quality function (modularity) calculates the difference between the original graph from its randomization. In such randomization, communities are eliminated, consequently, the comparison among the original structure and its randomization uncovers how non-random the cluster structure is (Coscia et al., 2011; Fortunato, 2010; Fortunato & Hric, 2016; Newman & Girvan, 2004).

k_i and k_j are the degrees of i and j, the likelihood p_i select at the random endpoint with i is $\frac{k_i}{2m}$, since there are k_i endpoint incidents with i out of a total of 2m. The likelihood of a link between i and j is the product $p_i p_j$ since edges are positioned independently of each other. The outcome is $\frac{k_i k_j}{4m^2}$, which delivers a predicted number $P_{ij} = 2mp_i p_j = \frac{k_i k_j}{2m}$ of edges between i and j (Fortunato, 2010). Thus, modularity expression becomes

$$Q = \frac{1}{2m}\sum_{ij}\left(A_{ij} - \frac{k_i k_j}{2m}\right)\delta(C_i, C_j)$$

Equation 5. Improved Modulariy Function (Fortunato, 2010; Fortunato & Hric, 2016)

4. Case Study

4.1. Test Data & Study Area

4.1.1. MVG Rad 2022 Bike-Share Mobility Data

The core of our case study lies in the MVG Rad Bike Sharing Mobility data (*MVG Rad*, 2023), big data stored in CSV (Comma-separated values) format. Each entry in this dataset represents a unique bicycle trip within the Munich area. These entries are structured with attributes separated by commas, and the dataset is encoded in UTF-8 for compatibility and readability.

Here is a glimpse of the data, with an illustrative example:

Table 2. Example of MVG Rad Bike-Sharing Mobility Data

	A	B	C	D	E	F	G	H	I	J	K	L
1	OID_	Row	STARTTIME	ENDTIME	STARTLAT	STARTLON	ENDLAT	ENDLON	RENTAL_IS_	RENTAL_STA	RETURN_IS_	RETURN_STA
2	1	1	1/1/2022 0:00	1/1/2022 0:00	48.13658	11.59283	48.14159	11.59721	0		0	
3	2	2	1/1/2022 0:00	1/1/2022 0:00	48.13659	11.59282	48.14165	11.59701	0		0	
4	3	3	1/1/2022 0:00	1/1/2022 0:00	48.16849	11.55028	48.15526	11.54012	0		1	Albrechtstrasse
5	4	4	1/1/2022 0:00	1/1/2022 0:00	48.16852	11.5502	48.15526	11.54012	0		1	Albrechtstrasse
6	5	6	1/1/2022 0:00	1/1/2022 0:00	48.16862	11.55234	48.15288	11.56669	0		0	
7	6	7	1/1/2022 0:00	1/1/2022 0:00	48.14159	11.59721	48.14255	11.57747	0		0	
8	7	8	1/1/2022 0:00	1/1/2022 0:00	48.14165	11.59701	48.13715	11.57639	0		0	

The MVG Rad system, owned by Munich's public transport provider MVG, represents a combination of station-based and free-floating bike-sharing systems. MVG, a subsidiary of Stadtwerke München GmbH, operates this service across an expansive area covering 110 square kilometers. MVG Rad is a significant component of Munich's transportation landscape, offering approximately 4,500 rental bikes adorned in the distinctive silver and blue MVG Bike livery (Albiński et al., 2018; *MVG Rad*, 2023). These bikes are available not only in Munich itself but also extend into nearby districts such as Starnberg and the municipality of Poing. Accessing these bikes is made easy through the official MVG website and the MVGO app. To utilize the service, users need to register (Albiński et al., 2018; *MVG Rad*, 2023). The app helps users locate the nearest MVG bike, facilitates bike reservation up to 15 minutes in advance, and issues a unique PIN for unlocking the bike directly from the onboard computer. Users can conveniently drop off their MVG bikes anywhere within the city's business area. In the surrounding districts, returns can be made only at designated bike stations. As an added incentive, anyone returning an MVG bike to a city station receives up to 5 free minutes as a token of appreciation (Albiński et al., 2018; *MVG Rad*, 2023).

The dataset (Table 2) captures various aspects of each bike trip, including bike IDs, geographic coordinates (latitude and longitude), station IDs (if the bike was parked at a station), and timestamps marking the journey's start and end.

4.1.2. Land Use & Population Data

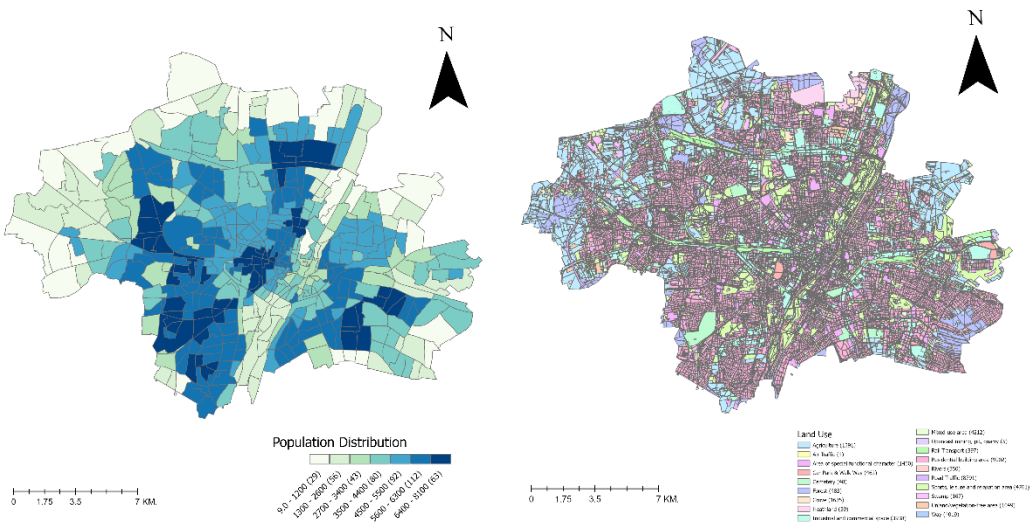


Figure 9. Population & Land Use Data projected on neighborhood level spatial unit of Munich

Addressing how semantic information is assigned to communities enables a richer understanding of the structures identified. By assigning land use & and population information to our communities, we have defined spatial characteristics. Thus, this semantic information aids in the interpretation of community behavior.

The population data was retrieved from https://data.humdata.org/dataset/kontur-population-germany and, is openly available under Creative Commons attribution international license. The dataset was built based on the Hexagonal Hierarchical Spatial Index. Calculation of population is based on overlapping Global Human Settlement Layer (GHSL) with Facebook High-Resolution Settlement Layer (HRSL) population data where available.

The land use data is part of ALKIS® and retrieved from https://www.ldbv.bayern.de/produkte/kataster/tat_nutzung.html. It describes the use of the earth's surface in four main groups (settlement, traffic, vegetation, and water). They divide these main groups into almost 140 different types of use such as residential building areas, road traffic, agriculture, or rivers enabling detailed evaluations and analyses of the use of the earth's surface. In Bavaria, the offices for digitization, broadband, and surveying (ÄDBV) are responsible for the collection of data. The current aerial photos from the surveying administration, data from the agricultural and forestry administration as well as on-site surveys as part of cadastral surveys serve as the basis for the recording.

4.1.3. Munich, Germany

Our case study centers on Munich, the capital and most populous city of the Free State of Bavaria, Germany. This metropolis serves as a compelling scene for our research, offering a rich economic dynamism, cultural diversity, and geographical significance. As of May 2022, Munich is home to approximately 1,578,132 inhabitants, making it the third-largest city in Germany, trailing only Berlin and Hamburg. Furthermore, it ranks as the 11th largest city within the European Union (*City of Munich*, n.d.). Munich is famous for its rich cultural heritage, architectural landmarks, sports events, exhibitions, and the world-renowned Oktoberfest, which is the largest Volksfest (beer festival and traveling

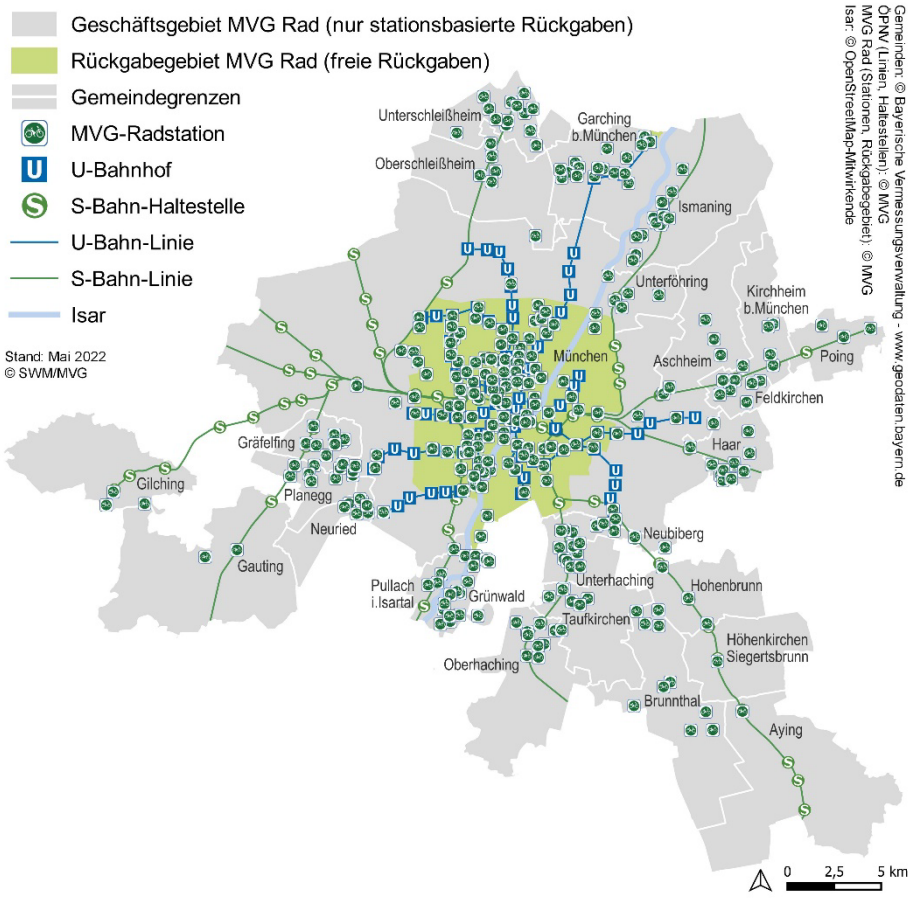


*Figure 10. MVG Bike Sharing Operation Area 2022 (*MVG Rad Karte*, 2022)*

funfair) in the world, attracting significant tourism. Geographically, Munich is situated in Upper Bavaria, about 50 kilometers north of the northern edge of the Alps. The city is characterized by the presence of the Isar and Würm rivers. It lies within the Northern Alpine Foreland, with its topography consisting of sandy plateaus, fertile flint areas, morainic hills, and fluvio-glacial out-wash fields (*City of Munich*, n.d.).

Cycling plays a significant role in Munich's transportation, accounting for 18% of all traffic in the city. Munich led the way in bicycle usage among major German cities, earning itself the title of "Germany's Bicycle Capital" in 2010. 80% of Munich's population owns a bicycle. From 1992 to 2010, the city invested €32 million in improving its cycling infrastructure, including bike parking facilities. Munich is committed to promoting cycling, aiming to increase its share to 20% of all trips by bike by 2015 (*Cycling in Munich*, n.d.). To achieve this goal, additional funding has been allocated for infrastructure development, public awareness campaigns, and community events. Starting in 2010, the annual budget for cycling promotion tripled to €4.5 million, underlining the city's dedication to sustainable transportation (*Cycling in Munich*, n.d.).

4.2. Dynamic Community Detection of Bike-Share Mobility Data

4.2.1. Data Preprocessing

Data preprocessing was a critical step in our project, primarily due to a significant number of errors (Figure 11) in the initial dataset. We conducted this preprocessing using ArcGIS Pro, and it involved several key stages to ensure data accuracy and relevance. Firstly, we imported district and traffic data into ArcGIS Pro. This formed the foundation of our data preparation process.

Next, we visualized the traffic data as a point layer, focusing on origin locations. This step helped us better understand the data's spatial distribution. To narrow down our analysis to the meaningful journey areas around central Munich, we used the "Select by Attribute" tools to choose origin locations in the vicinity. Ensuring that our data remained within the designated operation area was essential. To achieve this, we employed the "Select by Location" tool, using both versions of district data and bicycle traffic data as input. This step allowed us to select only the origin locations within the operation area. Following that, we conducted a reverse select process to remove origin locations that fell outside the operation area, further refining our dataset.

To validate the consistency of destination locations, we visualized them (Figure 12) and addressed any remaining errors in the data. We repeated the selection and validation process for destination locations to ensure data consistency across the data. To focus our analysis on weekdays, we separated the data into weekday and weekend categories, considering only the data from weekdays. Through the application of GIS tools such as "Select by Location" and "Select by Attribute," we performed data cleaning, significantly reducing the total number of trips from an initial 709,000 to 485,000. Finally, we employed spatial join operations to assign land use information to the respective spatial districts, enriching our dataset with additional contextual information.

4.2.2. Static Community Detection on Snapshots

To detect static communities, we adapted existing Python code and applied it to our data using Jupyter Notebook. We constructed graphs with nodes representing spatial units and edges representing traffic flows between districts. The edge weights were determined by counting the number of bicycle trips between district pairs.

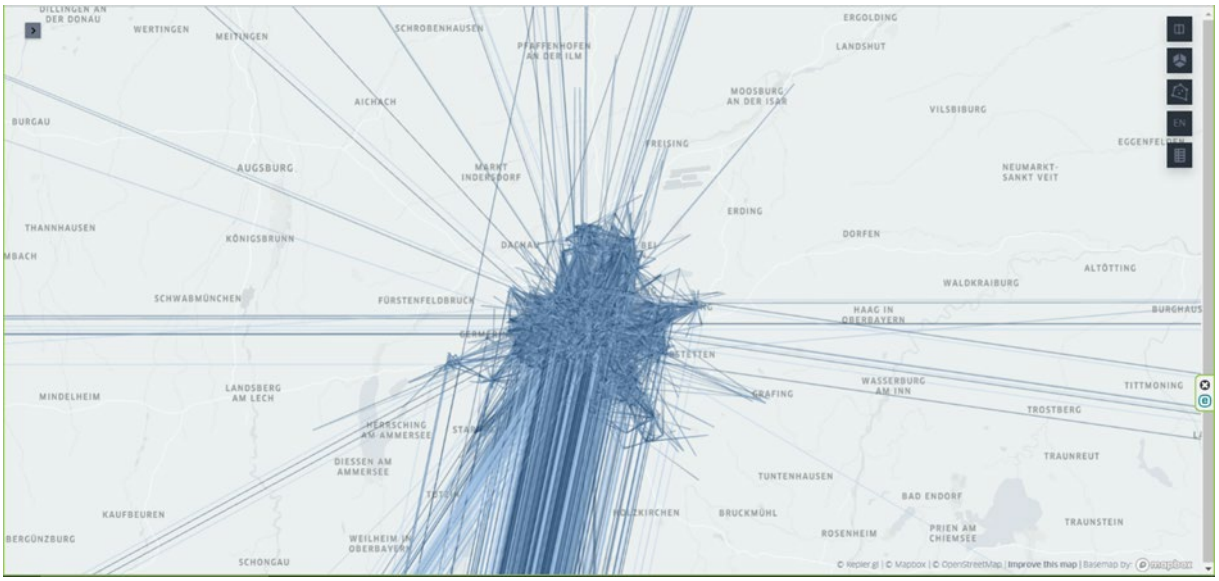


Figure 11. Visualization of Raw Mobility Data via Kepler.gl

The preliminary step for detecting dynamic communities is to transform our data into a mobility network. As stated in the previous chapter, to be able to create a network one should generate an adjacency (OD) matrix to map traffic flows between districts. To be able to identify which spatial unit each origin and destination location belongs to we need to apply a spatial match. The "sjoin()" function from the Geopandas library is used for this step. The function matches each entry of two input tables based on the geometry information. As geometry information, we have coordinates of each OD location and also districts of Munich. Since we would like to keep every bicycle trip and assign corresponding district index to them, we have applied "left join" where the bicycle trip data is kept as it is and spatial matches assigned from district data. Additionally, due to having two coordinate sets as OD for each bicycle trip, spatial matches are done separately for origins and destinations.

To be able to detect communities in our bicycle network, first, we need to create a graph that consists of nodes and edges. At this point, as we discussed, nodes are our spatial units with no weights, and, edges are traffic flows between districts with weights as the number of trips. To assign these bicycle trips as weights to our edges, we need to define possible trip directions between districts (from their index) and then we need to count how many trips occurred between each district pair. This is achieved with adjacency matrix A, where A(i,j) records the weight of the edge from node i to node j. The size of the matrix is defined with the square of the district count.

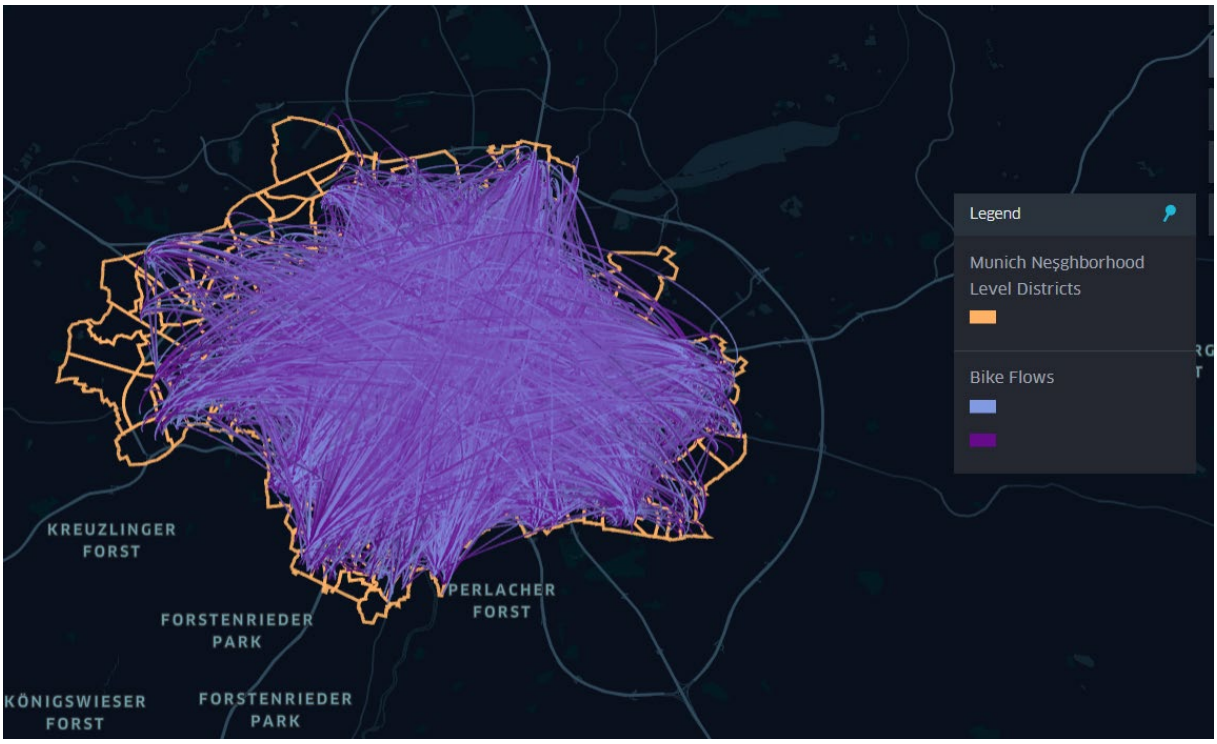


Figure 12. Data After Pre-Processing via kepler.gl

First, we have created a zero matrix by utilizing the “numpy” libraries’s “zeros” function. We have passed the total number of spatial units as dimensions to the matrix. Since we have a total number of 475 districts, the size of the matrix became 475x475. Then, entries of two tables which we have generated after spatial match, are iterated to get the spatial unit index of origin and destination locations for each trip, respectively. On each iteration, the code matches the origin and destination indexes for a trip, it passes this information to the adjacency matrix and, increases the corresponding element by one. For example, if a bicycle trip occurred from District 1 to District 2, the algorithm increases the corresponding element A_{12} of the matrix A by one. To be able to process bike-sharing flows, the algorithm converts the matrix into 3 column OD list where columns are “Origin”, ”Destination”, and ”Weight”.

In the next step, we have utilized the “DiGraph()” function from Pythons’s “netwokx” library. The DiGraph() function initializes a directed graph object that can store nodes and edges with optional attributes. After initializing the graph, the algorithm iterates over each entry of the OD list, adds an edge, and the weight of the edge is assigned using the corresponding value from the 'weight' column of the OD list. After assigning the edges, the DiGraph function automatically generates nodes from the endpoint information of the edges.

In the last step of this section, the InfoMap algorithm is implemented to detect bike-sharing communities. Python's "InfoMap" library was utilized for this step. The working principle of the algorithm is defined in section 3.2. In our code, we simply initialize an InfoMap instance, and load our mobility network into this instance. By implementing the "infomap.run()" function, the algorithm calculates communities within the given network. Then, the output of the infomap algorithm is formatted into a CSV file which has two columns as "Node" and "Community". In this way, we were able to identify which district belongs to which community at the end. The community assignment table of each snapshot is then aggregated into one shapefile for visualization and analysis purposes.

4.2.3. Dynamic Community Detection Throughout the day

To capture dynamic community structures, we built a consensus graph based on community membership graphs (CMGs). We iteratively updated the consensus matrix (CM) to reflect the strength of connections between edges across 12 snapshots. This allowed us to assess how strongly a community is connected throughout the day.

The first step of the dynamic community detection is to generate community membership graphs using the time series of static communities $\{C^1, \dots, C^t, \dots, C^k\}$, detected in the first stage. For each detected community $C_i^t \in C^t$, a community membership graph is constructed by connecting an undirected edge between any two nodes within the same community.

The algorithm iterates over nodes of detected static communities of each snapshot and generates an empty community membership graph consisting of community nodes but without edges. Then, it does a pairwise conditional check if the two nodes are in the same community. If the condition is met, then an edge is added to the corresponding community membership graph. The resulting graph is undirected and unweighted.

In the second step, we have created a consensus graph (referred to as "GC") that represents the dynamic community structure of a mobility network. The consensus graph will reveal how strongly edges are connected throughout multiple snapshots.

To start, an empty consensus matrix, denoted as "CM" is initialized. This matrix stores information about the connections between nodes (or edges) across different time snapshots. The code iterates over the set of community membership graphs (CMGs). Each CMG represents the community structure of the network at a specific time snapshot. Within each CMG, the code checks if an edge (connection) exists between nodes. If an edge exists in the CMG, it updates the corresponding element in the consensus matrix "CM" by adding "1" to that element. This update reflects the presence of the edge at that particular time snapshot. Here, the maximum value of

an element of CM can be 12 since we have 12 snapshots. This process allows us to consider how strongly an edge is connected throughout time.

Finally, we have applied the InfoMap algorithm to the consensus network GC. As a result, we have defined dynamic communities of bike-sharing usage within the city of Munich. The resulting communities reflect the parts where dense connections occured both spatially and temporally.

4.2.4. Dynamic Sub-Community Detection

The final stage of the workflow that we have proposed is the detection of dynamic sub-communities to reveal daily mobility patterns. Since we have generated strongly connected consensus communities, now we can find community shifts within each community. To be able to reveal sub-communities, we have generated separate graphs from each detected consensus community. First, an empty dictionary object is generated to store dynamic graphs in the following steps. Next a nested for loop, iterates on nodes of the consensus networks “GC” to collect the nodes with the same community assignment and it generates empty dynamic graphs such as “GD1, .., GDi” to store community nodes accordingly. Similar to section 4.2.2. first, we divided our mobility data into 48 intervals of 30 minutes, then OD matrixes for each community and each snapshot is generated. Edges that occur within certain communities and snapshots are added to the corresponding graph. In the following step, the infomap algorithm is applied to all dynamic graphs that are generated. Since we have revealed 6 consensus communities above GC, and have 48 snapshots, the infomap algorithm has applied to 288 graphs separately to reveal sub-communities.

4.3. Summary

Our study involves dynamic community detection of bike-sharing mobility data, focusing on data preprocessing and community detection. Data preprocessing involves cleaning and matching spatial information. For community detection, we employ static and dynamic approaches. The static approach involves constructing a mobility network, generating an adjacency matrix to map traffic flows, and applying the InfoMap algorithm for community detection. In the dynamic approach, we build consensus graphs based on community membership graphs (CMGs) across 12 snapshots to assess the strength of connections between edges throughout the day. Subsequently, dynamic graphs are derived from consensus communities to identify shifting communities throughout the day. This comprehensive methodology enables us to identify dynamic communities within Munich's bike-sharing network, reflecting both spatial and temporal patterns of usage.

5. Results & Discussion

In this chapter, the proposed dynamic community detection workflow is evaluated based on results and quality metrics.

Static Communities of Munich's Bike-Sharing System

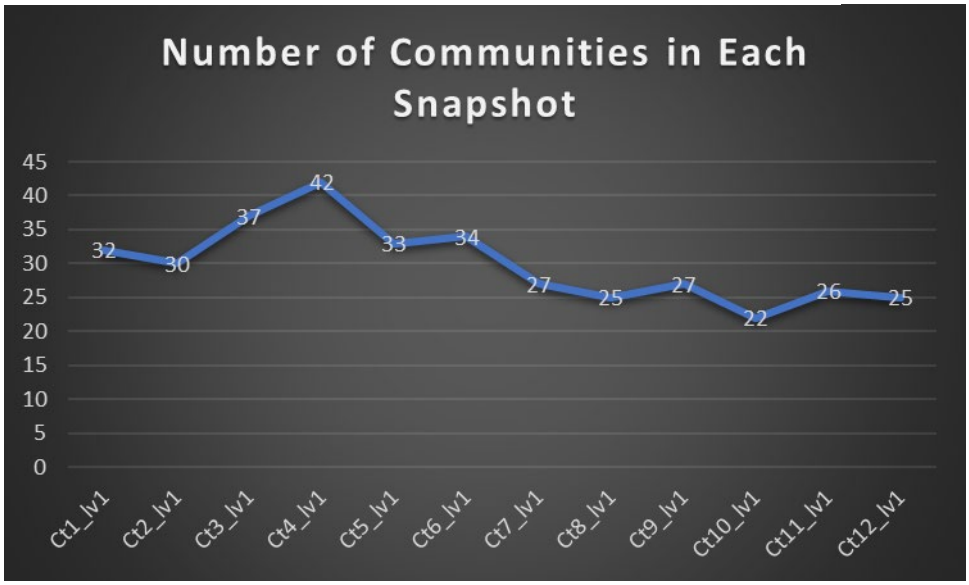
As a result of static community detection, we can see that communities in different time slots vary significantly. A total number of communities within each snapshot is used to quantify varying temporal characteristics of static communities (Table 3). It can be seen that the number of communities significantly increases from interval T2 (02:00 – 04:00) to interval T4 (06:00 – 08:00). These findings also align with Figure 6 and Table 3 where bike-sharing communities change significantly throughout the time.

Figure 13. Static Community Detection on Each Snapshot

Since the data only consists of bike trips that occurred on weekdays, we can see from snapshots T4 and T9 (06:00 – 08:00, 16:00 – 18:00) that there are increase in the number of communities before, and a significant drop after the rush hour periods. In addition, we can see a consistent distribution of communities during afternoon and evening times (Figure 13). The findings of the static community detection

algorithm reveal that even though it is good at finding partitions on a certain time interval, we can see that from Figure 13, human mobility is dynamic and one should consider temporal aspects of human mobility while finding meaningful clusters within a mobility network.

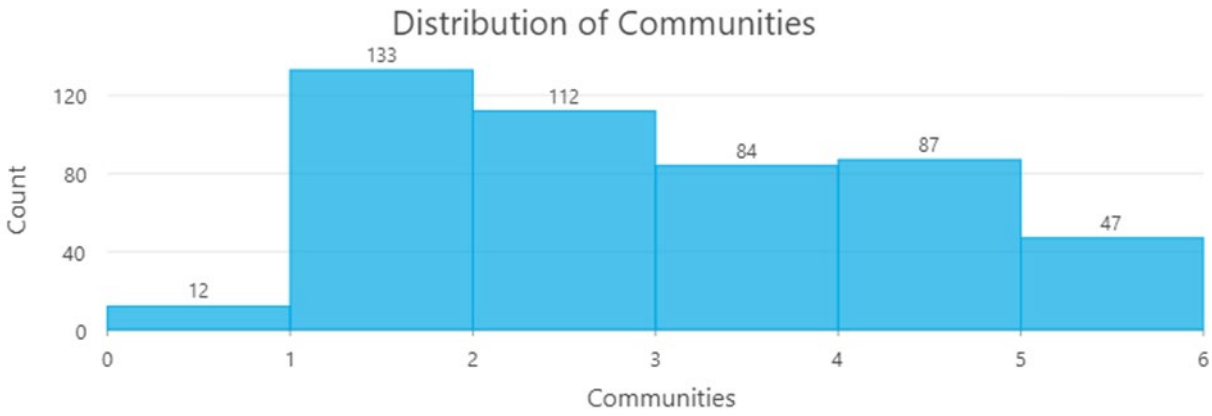
Table 3. Number of Communities within each 2h time interval



Dynamic Community Detection Throughout the Day

In the next step, detected static communities utilized to generate consensus graph GC, and by applying the infomap algorithm we have found both spatially and temporally, densely connected dynamic communities of bike-sharing mobility in Munich. As a result of dynamic community detection, 6 communities have been defined within the study area. It can be seen from Figure 14 that detected dynamic communities show good spatial clustering patterns. Moreover, we have applied the modularity function to evaluate how good the partition of the network is. We have calculated the modularity function of consensus communities and found that Q = 0.4786. This result indicates that the resulting partition significantly differs from a random graph and shows good clustering. The calculated Q value also aligns with the visualization of the consensus communities. However, there are certain parts of the graph where communities are spatially discontinuous. In the eastern part of the city, some districts are connected to communities 1 and 2. These regions seem to mostly consist of green space and a small portion of residential area. We can understand that people who live in this area tend to travel mostly to the city center. Since the data consists of weekday travel data, we don’t expect to see leisure-related trips from or to the city area.

Table 4. Number of nodes (Spatial Units) within each community



In addition, we can see from Figure 16 that the borders of the community partitions show certain patterns just in the city center area. To illustrate, the border of the community 1 and 2 display parallelity with the Central Train Station of Munich. The community border between 1 – 4, 2 -4, and 2 – 6 displays parallelity along the river Isar. The border between communities 4 and 6 splits over the intersection of the train line and highway. This pattern is also similar between communities 1 and 3 which is divided by a highway. So we can conclude from visual analysis of the dynamic communities that, bike-sharing usage is highly relevant to the built environment. People can not overcome physical barriers such as train lines, rivers, and highways and tend to travel between regions covered by these natural and artificial barriers. Therefore, RQ 1.2.2. and RQ 1.3.1. can be answered as follows; combining community structures with built environment information indeed provides us with understanding regarding human mobility and travel patterns.

Additionally, to be able to identify characteristics we have assigned land use information to detect dynamic communities. The following Figure 15 shows the percentage of areal coverage of land use information within the study area. Community 1 mainly consist of residential area and is followed by industrial/commercial areas.

For Community 2, the region mainly consists of residential characteristics but unlike Community 1, residential areas are followed by recreational and leisure-related areas. Community 3 mainly consists of residential areas followed by areas with agricultural functionality. Community 4 includes industrial and leisure-related functionality after residential areas. For community 5 second main land use attribute forest regions while for community 6, the residential area covers almost the whole community area. We can conclude from the findings that each community represents distinct characteristics, bike share users have different travel purposes among communities. So, we can define travel purposes with the community patterns and these findings also answer the RQ 1.3.1.

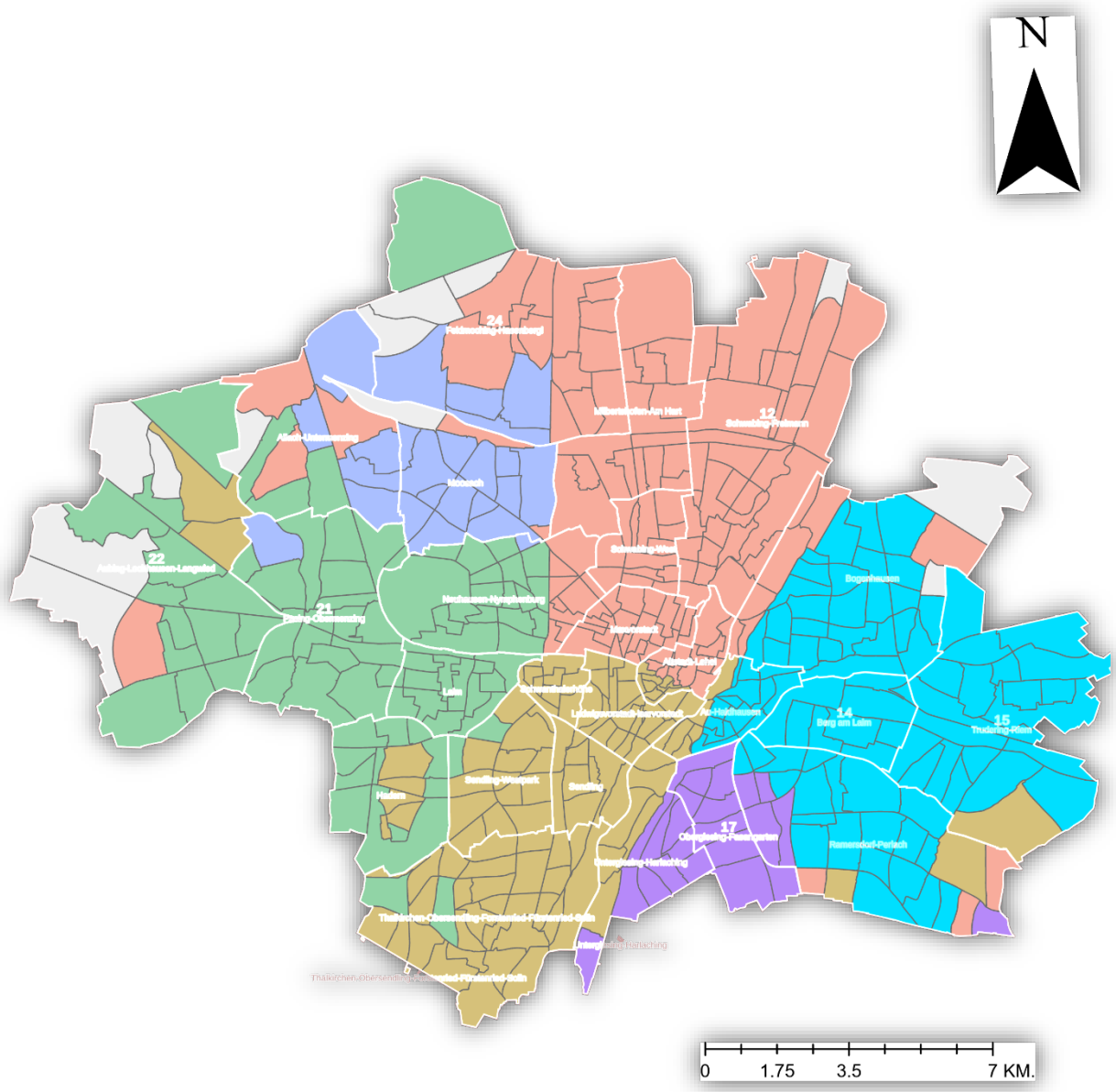


Figure 14. Visualization of Dynamic Communities around the city center area

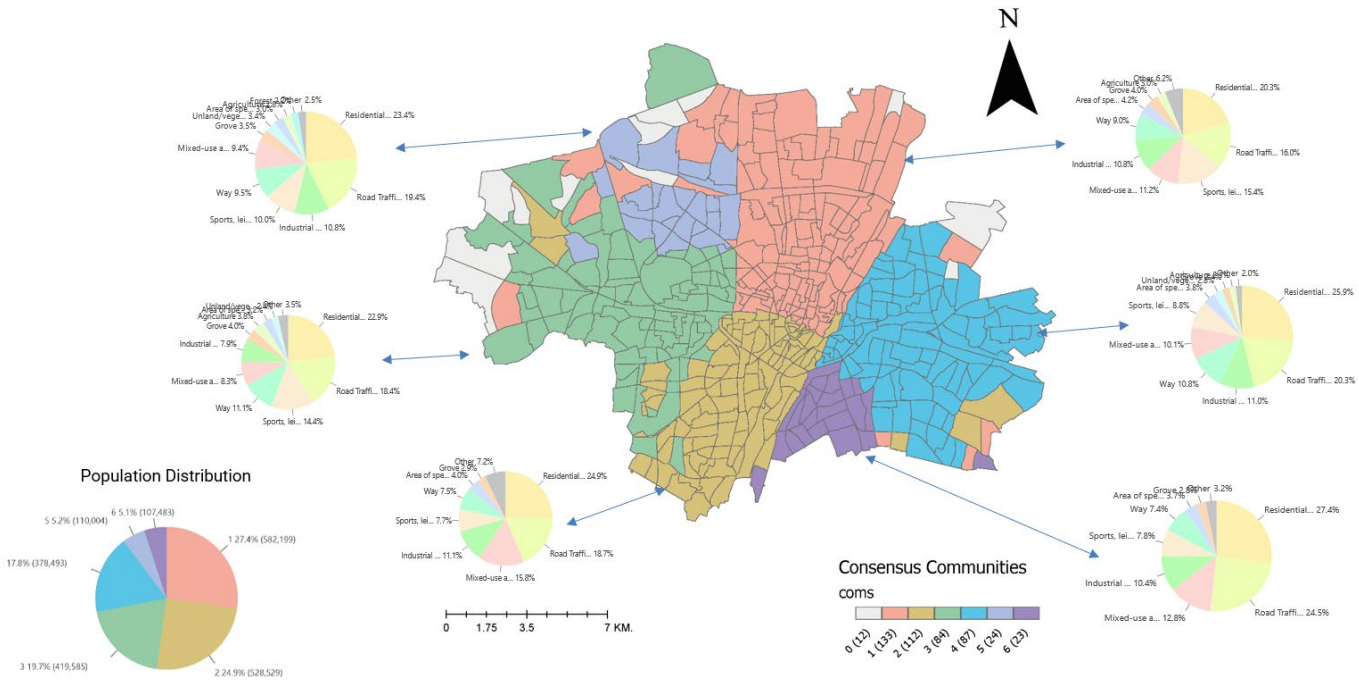


Figure 15. Resulting Dynamic Communities of Munich with land use and population distribution information.

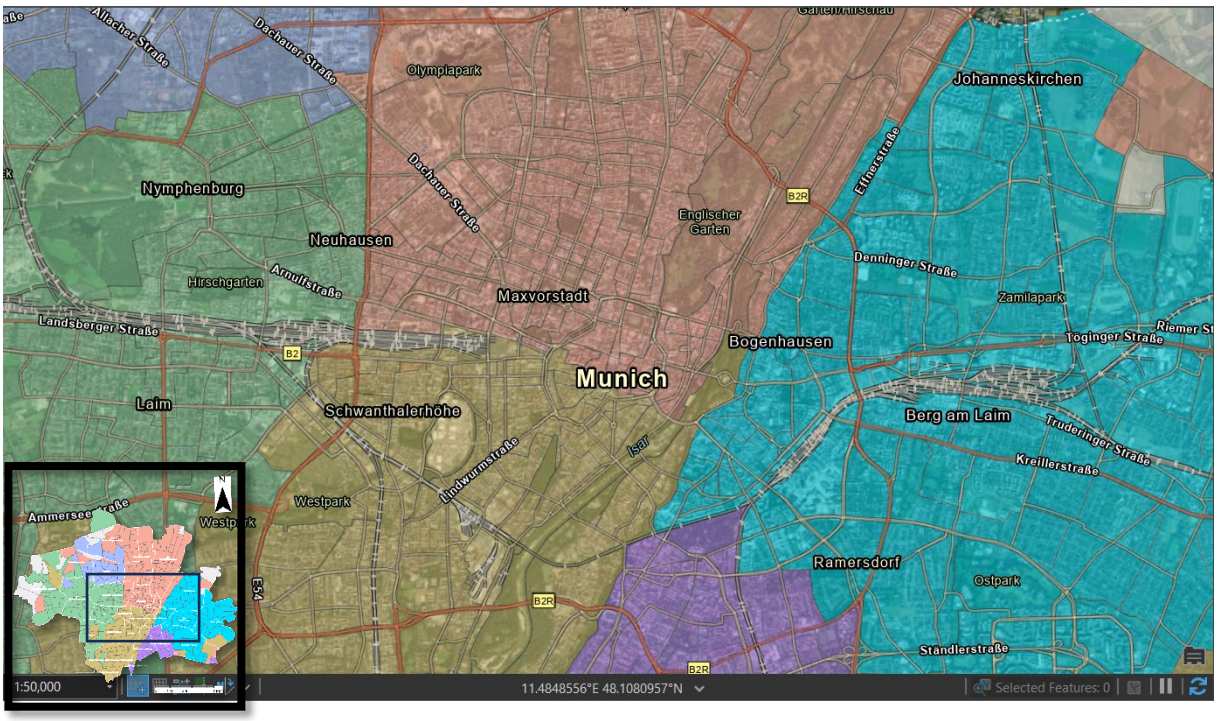


Figure 16. Visualization of Dynamic Communities around the city center area

Dynamic Sub-Communities

In the final stage, we have defined dynamic sub-communities throughout the day (Figure 15) and revealed the daily rhythms of human mobility. The detected communities allow us to understand human mobility patterns in the context of bike-sharing systems. We have evaluated our results under two different scales that are global & local.

- Global Scale

It can be seen from the results (Figure 17 & Table 5) that the number of connected spatial units within each dynamic community changes significantly in short time intervals. To illustrate, within Community 1, there are 12 connected regions around 06:00 AM (T12) in the morning, and this number drastically increases to 84 around 09:00 AM (T18). After morning rush hour, the number of connected units fluctuates throughout the day as it decreases to 57 around 11:00 AM (T22) and, in the evening rush hours it again increases to 85 at 17:30 (T35) while it reaches its daily peak to 86 at 19:00 (T38) (Table 4) This pattern shows parallelity among all communities which reveal humans travel behaviors within the city of Munich.

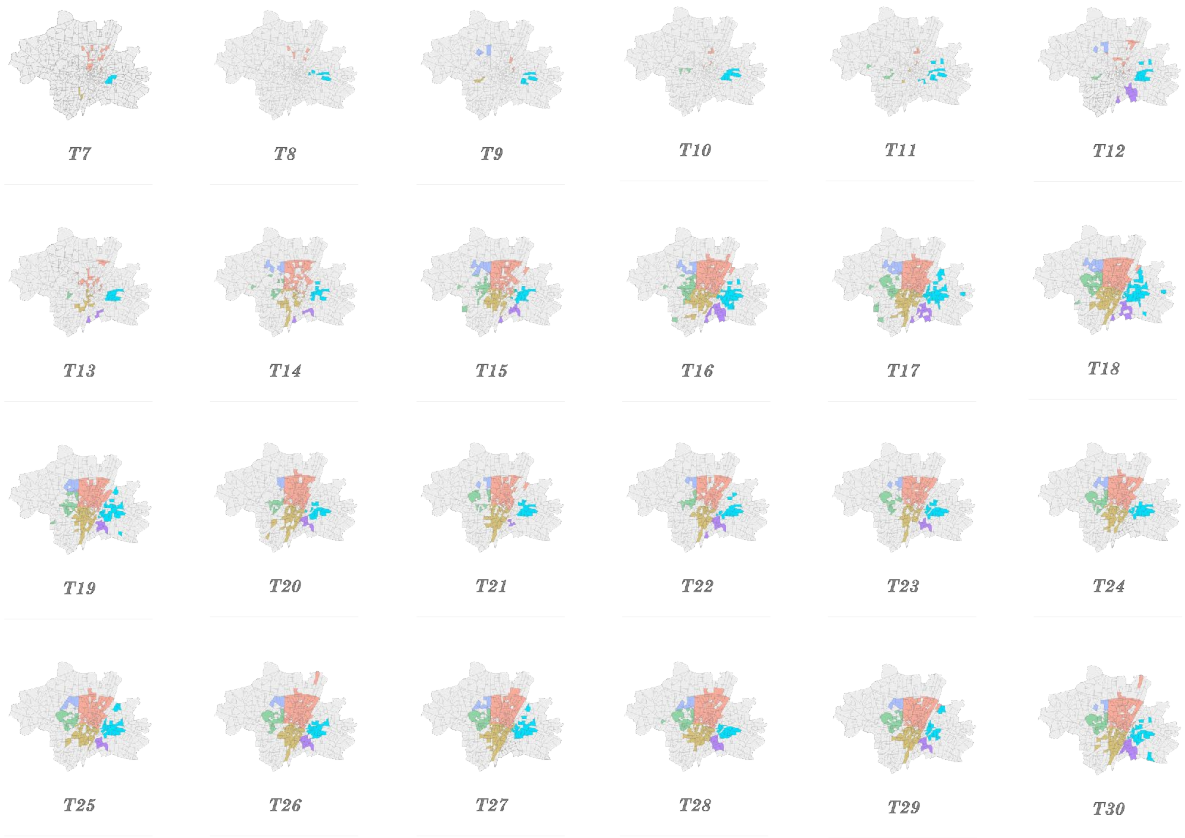
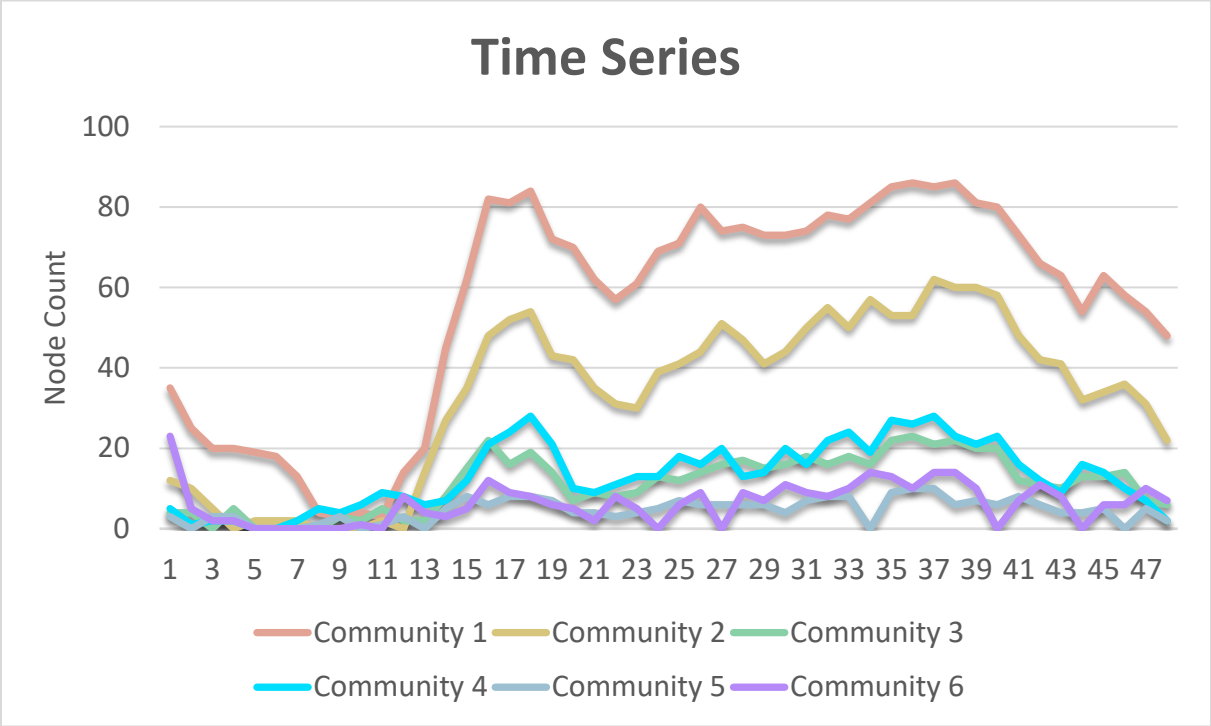


Figure 17. Dynamic Sub-Communities Throughout the Day (Between T7 and T30)

Table 5. Time Series of Node Counts within Detected Consensus Communities



Moreover, certain regions stay connected throughout the day. For example, in community 1 Olympiapark & Maxvorstadt, community 2 Ludvigsvorstadt & Untergiesing, community 3 Laim & Nymphenburg, community 4 Berg am Laim & Zamdorf, community 5 Moosach & Olympia – Einkaufzentrum, and community 6 Stadelheim & Giesing. There is either a tram, train, or a subway station in the middle of these regions. We can conclude that on a global scale, bike-share users prefer to use bicycles mainly as a complementary mode to public transportation and for commuting purposes during the weekdays.

Dynamic Sub-Communities - Local Scale

To illustrate the significance of the detecting dynamic sub-communities we have focused on Community 1 in this section. Evaluating each spatial unit with underlying land use information allowed us to reveal travel purposes during certain time intervals.

Figures 18 and 19 show sub-communities within Community 1 during time intervals of T13 to T16. We have identified that certain regions stay connected during rush hours. For example, the spatial unit that includes Olympiadorf and Olympiazentrum subway station stays connected to the spatial unit that includes Hochschule Mücnhen and the Northern part of the Milbertshofen that includes industrial amenities such as BMW campuses. In addition, spatial units of Oberföhring and Schwabing fall into the same community during rush hours. In the center of the spatial unit of Oberföhring, there is Grundschule, and Schwabing mainly consists of

residential settlements. We can understand that people in these regions are mainly using these bicycles for commuting and they are either traveling to their job or their school of education. Considering the fact that students are the main residents of the Olympiadorf, they were using these bicycles to regions without subway or tram stations and the purpose of these trips might be to go to their internship or their lectures.

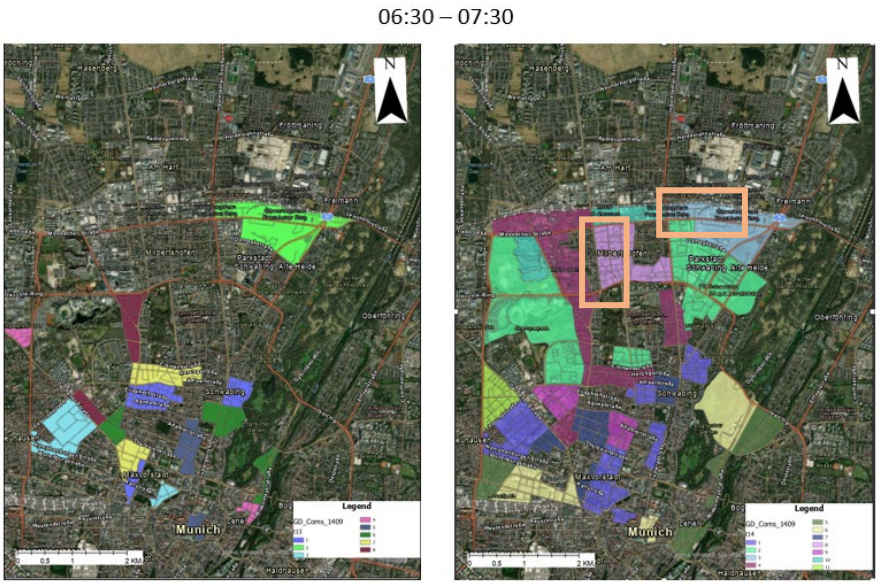


Figure 18. Detected Sub-Communities of Community 1 between 06:30 – 07:30. Orange boxes refer to Olympiadorf on the west, and BMW Campuss on the Northern part.

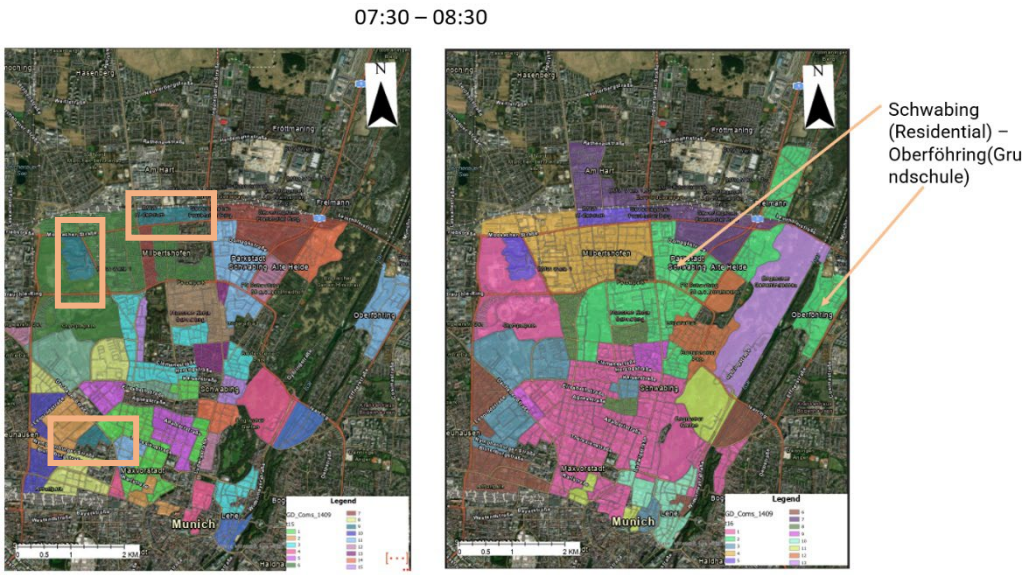


Figure 19. Detected Sub-Communities of Community 1 between 07:30 – 08:30.

5.1. Discussion

The objectives and research questions of this study have been successfully addressed through the development of a comprehensive spatiotemporal analysis workflow for modeling dynamic community structures of hybrid bike share usage.

RQ 1.1.1: By investigating the current spatial pattern analysis methods of hybrid bike share system usage, we have gained insights into the existing approaches for understanding how these systems are utilized. This evaluation has informed our subsequent analyses.

RQ 1.2.1: Determining the most suitable temporal time unit for the adopted method was essential for capturing meaningful spatiotemporal patterns. This choice has been made to ensure the accuracy and relevance of our dynamic community detection.

RQ 1.2.2: Identifying additional attributes or parameters such as community detection algorithms and modularity functions that could enhance our network analysis allowed us to extract more meaningful information. These enhancements have improved the depth of our community detection.

RQ 1.2.3: Addressing how semantic information is assigned to communities enables a richer understanding of the structures identified. By assigning land use information to our communities, we have defined spatial characteristics. Thus, this semantic information aids in the interpretation of community behavior.

RQ 1.3.1: The application of community detection methods has proven to be valuable in extracting travel purposes, shedding light on why and how hybrid bike share systems are used within urban areas. This contributes to urban transport planning.

RQ 1.3.2: Identifying dynamic communities offers insights into both spatially and temporally connected clusters of bike share usage over time. This understanding is crucial for adapting urban transport planning strategies to meet planning needs.

In summary, by addressing these research questions within the context of their respective sub-objectives, we have successfully achieved the overarching objective of developing a spatiotemporal analysis workflow for modeling dynamic community structures of hybrid bike share usage. This workflow can be applied to Urban Transport Planning and offers valuable insights into travel patterns and community behavior within hybrid bike-share systems.

5.2. Limitations & Future Work

The developed workflow and algorithm allow us to gain insights about community structure and according to quality metrics, it performed well. However, the algorithm has many sub-steps for manipulating the big data which reduces the efficiency of computation. Moreover, in these sub-steps, only spatial district indexes are taken into account. This means that we extracted the spatial units and converted them to topological space. Then, after finding communities in topological space, we converted this information into geographical space by assigning community labels to spatial units. Therefore, future studies should also focus on how we can conduct community detection algorithms in geographical space. In this way can compute more spatial parameters along with temporal aspects such as distance or elevation.

6. Conclusion

In this research, we have presented a comprehensive methodology for the detection and analysis of bike-sharing communities within mobility networks. We began by preprocessing the data, selecting relevant parameters, and cleaning the datasets to ensure data consistency and reliability. Subsequently, we applied static community detection using the InfoMap algorithm to identify communities at different time snapshots, shedding light on temporal patterns in bike-sharing usage. Furthermore, we introduced dynamic community detection by developing a consensus clustering method that identified evolving communities throughout the day. This approach allowed us to uncover and analyze the dynamic nature of bike-sharing communities, providing valuable insights into the complex patterns of human mobility.

To evaluate the quality of our community detection results, we employed the modularity quality criterion, which quantifies the effectiveness of the clustering process. Modularity assesses both the internal cohesion of communities and the absence of edges between communities, helping us determine the quality of our detected communities.

In conclusion, this research has provided a robust methodology for the analysis of bike-sharing communities, considering both spatial and temporal aspects. By applying dynamic community detection to mobility data, we can better understand how these communities evolve throughout the day and gain valuable insights into urban mobility patterns. This methodology has the potential to inform urban transport planning and contribute to our understanding of how bike-sharing systems can be optimized for sustainable and efficient transportation in cities. Overall, the study contributes to the fields of Geographic Information Science, Cartography, and Urban Transport Planning by offering a systematic approach to analyzing human mobility patterns within bike-sharing systems, with a focus on dynamic community structures. Further research and applications in this area could lead to more effective and sustainable urban transportation solutions.

Bibliography

Albiński, S., Fontaine, P., & Minner, S. (2018). Performance analysis of a hybrid bike sharing system: A service-level-based approach under censored demand observations. *Transportation Research Part E: Logistics and Transportation Review*, *116*, 59–69. https://doi.org/10.1016/j.tre.2018.05.011

Bachand-Marleau, J., Lee, B. H. Y., & El-Geneidy, A. M. (2012). Better Understanding of Factors Influencing Likelihood of Using Shared Bicycle Systems and Frequency of Use. *Transportation Research Record: Journal of the Transportation Research Board*, *2314*(1), 66–71. https://doi.org/10.3141/2314-09

Barthélemy, M. (2011). Spatial networks. *Physics Reports*, *499*(1–3), 1–101. https://doi.org/10.1016/j.physrep.2010.11.002

Bauman, A., Crane, M., Drayton, B. A., & Titze, S. (2017). The unrealised potential of bike share schemes to influence population physical activity levels – A narrative review. *Preventive Medicine*, *103*, S7–S14. https://doi.org/10.1016/j.ypmed.2017.02.015

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., & Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, *2008*(10), P10008. https://doi.org/10.1088/1742-5468/2008/10/P10008

Chen, W., Chen, X., Cheng, L., Liu, X., & Chen, J. (2022). Delineating borders of urban activity zones with free-floating bike sharing spatial interaction network. *Journal of Transport Geography*, *104*, 103442. https://doi.org/10.1016/j.jtrangeo.2022.103442

Chen, Z., Van Lierop, D., & Ettema, D. (2020). Dockless bike-sharing systems: What are the implications? *Transport Reviews*, *40*(3), 333–353. https://doi.org/10.1080/01441647.2019.1710306

City of Munich. (n.d.). [Wikipedia]. https://en.wikipedia.org/wiki/Munich

Coscia, M., Giannotti, F., & Pedreschi, D. (2011). A classification for community discovery methods in complex networks. *Statistical Analysis and Data Mining*, *4*(5), 512–546. https://doi.org/10.1002/sam.10133

Cycling in Munich. (n.d.). [Wikipedia]. https://en.wikipedia.org/wiki/Cycling_in_Munich

Danon, L., Díaz-Guilera, A., Duch, J., & Arenas, A. (2005). Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*, *2005*(09), P09008–P09008. https://doi.org/10.1088/1742-5468/2005/09/P09008

Dao, V.-L., Bothorel, C., & Lenca, P. (2021). An empirical characterization of community structures in complex networks using a bivariate map of quality metrics. *Social Network Analysis and Mining*, *11*(1), 37. https://doi.org/10.1007/s13278-021-00743-1

DeMaio, P. (2009). Bike-sharing: History, Impacts, Models of Provision, and Future. *Journal of Public Transportation*, *12*(4), 41–56. https://doi.org/10.5038/2375-0901.12.4.3

El-Assi, W., Salah Mahmoud, M., & Nurul Habib, K. (2017). Effects of built environment and weather on bike sharing demand: A station level analysis of commercial bike sharing in Toronto. *Transportation*, *44*(3), 589–613. https://doi.org/10.1007/s11116-015-9669-z

Eren, E., & Uz, V. E. (2020). A review on bike-sharing: The factors affecting bike-sharing demand. *Sustainable Cities and Society*, *54*, 101882.

Euler, L. (1736). Solutio problematis ad geometriam situs pertinentis. *Comment Acad Sci Imperialis Petropolit*, *8*, 128–140.

Euler, L. (1953). Leonhard Euler and the Koenigsberg Bridges. *Scientific American*, *189*, 66–70.

Fan, A., Chen, X., & Wan, T. (2019). How Have Travelers Changed Mode Choices for First/Last Mile Trips after the Introduction of Bicycle-Sharing Systems: An Empirical Study in Beijing, China. *Journal of Advanced Transportation*, *2019*, 1–16. https://doi.org/10.1155/2019/5426080

Fishman, E. (2016). Bikeshare: A Review of Recent Literature. *Transport Reviews*, *36*(1), 92–113. https://doi.org/10.1080/01441647.2015.1033036

Fishman, E., Washington, S., Haworth, N., & Watson, A. (2015). Factors influencing bike share membership: An analysis of Melbourne and Brisbane. *Transportation Research Part A: Policy and Practice*, *71*, 17–30. https://doi.org/10.1016/j.tra.2014.10.021

Fortunato, S. (2010). Community detection in graphs. *Physics Reports*, *486*(3–5), 75–174. https://doi.org/10.1016/j.physrep.2009.11.002

Fortunato, S., & Hric, D. (2016). Community detection in networks: A user guide. *Physics Reports*, *659*, 1–44. https://doi.org/10.1016/j.physrep.2016.09.002

Gribkovskaia, I., Halskau, Ø., & Laporte, G. (2007). The bridges of Königsberg—A historical perspective. *Networks*, *49*(3), 199–203. https://doi.org/10.1002/net.20159

Ji, Y., Ma, X., He, M., Jin, Y., & Yuan, Y. (2020). Comparison of usage regularity and its determinants between docked and dockless bike-sharing systems: A case study in

Nanjing, China. *Journal of Cleaner Production*, *255*, 120110. https://doi.org/10.1016/j.jclepro.2020.120110

Kou, Z., & Cai, H. (2021). Comparing the performance of different types of bike share systems. *Transportation Research Part D: Transport and Environment*, *94*, 102823. https://doi.org/10.1016/j.trd.2021.102823

Kulikov, A. S. (n.d.). *Introduction to Graph Theory by University of California San Diego* [PDF File]. Retrieved July 1, 2023, from https://www.coursera.org/learn/graphs

Lancichinetti, A., & Fortunato, S. (2012). Consensus clustering in complex networks. *Scientific Reports*, *2*(1), 336. https://doi.org/10.1038/srep00336

Liao, L., Li, B., Huang, D., Xiao, Z., & Zheng, Q. (2023). An Intelligent Rebalance System for Tidal Phenomenon of Dockless Bicycle-Sharing. *IEEE Access*, *11*, 12937–12948. https://doi.org/10.1109/ACCESS.2023.3241799

Lin, P., Weng, J., Hu, S., Alivanistos, D., Li, X., & Yin, B. (2020). Revealing Spatio-Temporal Patterns and Influencing Factors of Dockless Bike Sharing Demand. *IEEE Access*, *8*, 66139–66149. https://doi.org/10.1109/ACCESS.2020.2985329

Lu, C., Gao, L., & Huang, Y. (2023). Exploring travel patterns and static rebalancing strategies for dockless bike-sharing systems from multi-source data: A framework and case study. *Transportation Letters*, *15*(4), 336–349. https://doi.org/10.1080/19427867.2022.2051798

McBain, C., & Caulfield, B. (2018). An analysis of the factors influencing journey time variation in the cork public bike system. *Sustainable Cities and Society*, *42*, 641–649. https://doi.org/10.1016/j.scs.2017.09.030

MVG Rad. (2023). https://www.mvg.de/services/mvg-rad.html

MVG Rad Karte. (2022). [Map]. SWM/MVG. https://www.mvg.de/.imaging/mte/mvg/galerie-view/dam/mvg/services/mobile-services/mvg-rad/MVG_Rad_Karte.jpg/jcr:content/MVG_Rad_Karte.jpg

Phillips, J. D., Schwanghart, W., & Heckmann, T. (2015). Graph theory in the geosciences. *Earth-Science Reviews*, *143*, 147–160. https://doi.org/10.1016/j.earscirev.2015.02.002

Reiss, S., & Bogenberger, K. (2015). GPS-Data Analysis of Munich's Free-Floating Bike Sharing System and Application of an Operator-based Relocation Strategy. *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, 584–589. https://doi.org/10.1109/ITSC.2015.102

Rixey, R. A. (2013). Station-Level Forecasting of Bikesharing Ridership: Station Network Effects in Three U.S. Systems. *Transportation Research Record: Journal of the Transportation Research Board*, *2387*(1), 46–55. https://doi.org/10.3141/2387-06

Rosvall, M., Axelsson, D., & Bergstrom, C. T. (2009). The map equation. *The European Physical Journal Special Topics*, *178*(1), 13–23. https://doi.org/10.1140/epjst/e2010-01179-1

Rosvall, M., & Bergstrom, C. T. (2008). Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, *105*(4), 1118–1123. https://doi.org/10.1073/pnas.0706851105

Shaheen, S. A., Guzman, S., & Zhang, H. (2010). Bikesharing in Europe, the Americas, and Asia: Past, Present, and Future. *Transportation Research Record: Journal of the Transportation Research Board*, *2143*(1), 159–167. https://doi.org/10.3141/2143-20

Shaheen, S. A., Guzman, S., & Zhang, H. (10/20212). Bikesharing across the Globe. In *City Cycling* (pp. 183–210). MIT Press. https://escholarship.org/uc/item/0qm296pf

Shen, Y., Zhang, X., & Zhao, J. (2018). Understanding the usage of dockless bike sharing in Singapore. *International Journal of Sustainable Transportation*, *12*(9), 686–700. https://doi.org/10.1080/15568318.2018.1429696

Shui, C. S., & Szeto, W. Y. (2020). A review of bicycle-sharing service planning problems. *Transportation Research Part C: Emerging Technologies*, *117*, 102648. https://doi.org/10.1016/j.trc.2020.102648

Song, J., Zhang, L., Qin, Z., & Ramli, M. A. (2021). A spatiotemporal dynamic analyses approach for dockless bike-share system. *Computers, Environment and Urban Systems*, *85*, 101566. https://doi.org/10.1016/j.compenvurbsys.2020.101566

The meddin bike-sharing world map report. (n.d.). [Sunum]. https://bikesharingworldmap.com/reports/bswm_mid2022re-port.pdf

Traud, A. L., Kelsic, E. D., Mucha, P. J., & Porter, M. A. (2011). Comparing Community Structure to Characteristics in Online Collegiate Social Networks. *SIAM Review*, *53*(3), 526–543. https://doi.org/10.1137/080734315

Tyler, J. R., Wilkinson, D. M., & Huberman, B. A. (2003). Email as Spectroscopy: Automated Discovery of Community Structure within Organizations. In M. Huysman, E. Wenger, & V. Wulf (Eds.), *Communities and Technologies* (pp. 81–96). Springer Netherlands. https://doi.org/10.1007/978-94-017-0115-0_5

Zhang, L., Zhang, J., Duan, Z., & Bryde, D. (2015). Sustainable bike-sharing systems: Characteristics and commonalities across cases in urban China. *Journal of Cleaner Production*, *97*, 124–133. https://doi.org/10.1016/j.jclepro.2014.04.006

Zhang, Y., Thomas, T., Brussel, M., & Van Maarseveen, M. (2017). Exploring the impact of built environment factors on the use of public bikes at bike stations: Case study in Zhongshan, China. *Journal of Transport Geography*, *58*, 59–70. https://doi.org/10.1016/j.jtrangeo.2016.11.014

Zhao, Y., Chen, B. Y., Gao, F., & Zhu, X. (2023). Dynamic community detection considering daily rhythms of human mobility. *Travel Behaviour and Society*, *31*, 209–222. https://doi.org/10.1016/j.tbs.2022.12.009

Zhou, X. (2015). Understanding Spatiotemporal Patterns of Biking Behavior by Analyzing Massive Bike Sharing Data in Chicago. *PLOS ONE*, *10*(10), e0137922. https://doi.org/10.1371/journal.pone.0137922

Appendix A – Python Code

Step 1: Static Community Detection on Snapshots

In this part, dynamic OD networks generated and community detection applied to each snapshot. The community results added to original shape file as "Ct1,...,Ct12".

In [ ]:
```
import glob
import pandas as pd
from datetime import datetime, timedelta
from geopandas import datasets, GeoDataFrame, read_file,sjoin #
from shapely.geometry import Point, Polygon, LineString
import geopandas
import numpy as np
import networkx as nx
import geoplot
import geoplot.crs as gcrs
import matplotlib.pyplot as plt
import os
os.chdir("C:/Users/yagiz/Desktop/Dynamic Com. Det.
Implementation/Data/Neighborhood")

path_g = "nl_lv/G_2h/"

all_od_networks = sorted(glob.glob(path_g + "*.csv"), key=os.path.getmtime)
print(all_od_networks)
# Read the shapefile "MVG_Operation_Area.shp" into a GeoDataFrame named 'cbg'
cbg = read_file("Neighborhood_Munich.shp")

# Convert the coordinate reference system (CRS) of 'cbg' to EPSG:4326. the shape
file originally has epsg:25832. but polyplot doesnt recognize it. threfore crs
conversion implemented.
cbg_new = cbg.to_crs('epsg:4326')
# Note: GeoPandas is converting the CRS to EPSG:4326 (WGS 84), which is commonly
used for latitudinal and longitudinal data.

#print(cbg_new)
#geoplot.polyplot(cbg_new, projection=gcrs.EuroPP(), edgecolor='blue',
facecolor='lightgrey', linewidth=.3, figsize=(12, 12))

for i in range(len(all_od_networks)):
    # Read the Snapshot graph CSV file "Gi.csv" into a DataFrame named 'df_info'
    df_info = pd.read_csv(all_od_networks[i])

# Print the number of rows (length) of the DataFrame 'df_info'
    print(len(df_info))

#print(df_info.head())

# Create Point geometries based on the 'STARTLON' and 'STARTLAT' columns of the
DataFrame 'df_info'
    geom1 = [Point(xy2) for xy2 in zip(df_info['STARTLON'], df_info['STARTLAT'])]

# Convert the DataFrame 'df_info' to a GeoDataFrame 'gdf_O' by adding the Point
geometries as the 'geometry' column
    gdf_O = geopandas.GeoDataFrame(df_info, geometry=geom1)

# Set the coordinate reference system (CRS) of 'gdf_O' to EPSG:4326 (WGS 84)
    gdf_O.crs = 'EPSG:4326'
```

Loading [MathJax]/extensions/Safe.js

```
    # Print the first few rows of the GeoDataFrame 'gdf_O' to inspect the data
    #print(gdf_O.head())

# Perform a spatial join between 'gdf_O' (left table) and 'cbg_new' (right table)
using a left join strategy.
# This will keep all entries (bike trips) from the left table and assign districts
from the right table.
    cbg_O = sjoin(gdf_O, cbg_new, how='left')

# cbg_O.head()


# Create Point geometries based on the 'ENDLON' and 'ENDLAT' columns of the
DataFrame 'df_info'
    geom2 = [Point(xy1) for xy1 in zip(df_info['ENDLON'], df_info['ENDLAT'])]

# Convert the DataFrame 'df_info' to a GeoDataFrame 'gdf_D' by adding the Point
geometries as the 'geometry' column
    gdf_D = geopandas.GeoDataFrame(df_info, geometry=geom2)

# Set the coordinate reference system (CRS) of 'gdf_D' to EPSG:4326 (WGS 84)
    gdf_D.crs = 'EPSG:4326'

# Perform a spatial join between 'gdf_D' (left table) and 'cbg_new' (right table)
using a left join strategy.
# This will keep all entries (bike trips) from the left table and assign districts
from the right table.
    cbg_D = sjoin(gdf_D, cbg_new, how='left')

# Remove duplicate rows based on the 'OID' column in 'cbg_O' and 'cbg_D'
GeoDataFrames
# This step is performed to handle cases where multiple points in the GeoDataFrame
correspond to the same district (polygon).
# This ensures that each district is represented only once in the final
GeoDataFrames.
    #cbg_O.drop_duplicates(subset='OID', inplace=True)
    #cbg_D.drop_duplicates(subset='OID', inplace=True)


# The head of 'cbg_D' is printed to inspect the first few rows of the GeoDataFrame
'cbg_D'.
#cbg_D.head()
    #pd.set_option('display.max_columns', None)
    #display(cbg_O)
    #display(cbg_D) #displays destination locations with corresponding district
indexes on the table.

    #cbg_O.plot()
    #print(cbg_D.plot())
    data_O=pd.DataFrame(cbg_O)
    data_D=pd.DataFrame(cbg_D)


# Create an OD matrix 'cbg_OD' initialized with zeros of size (len(cbg_new),
len(cbg_new))
    cbg_OD = np.zeros((len(cbg_new), len(cbg_new)))

# Loop through the rows of the DataFrame 'data_D'
    for j in range(len(data_D)):
    # Get the index_right value from 'data_O' and 'data_D' (corresponding to the
spatial join)
```

```
            p = data_O.index_right[j]
            q = data_D.index_right[j]

            if np.isnan(q):
                q = p
            elif np.isnan(p):
                p = q

        # Increment the corresponding element in the OD matrix 'cbg_OD' by 1
            cbg_OD[int(p)][int(q)] = cbg_OD[int(p)][int(q)] + 1

    # The commented-out line saves the OD matrix 'cbg_OD' to a CSV file.

    # Print the OD matrix 'cbg_OD' after the operations.
        #print(cbg_OD)

    # Create an OD weights array 'odw' initialized with zeros of size (len(cbg) *
    len(cbg), 3)
        odw = np.zeros((len(cbg) * len(cbg), 3))

    # Loop through all origins and destinations to populate the 'odw' array
        for k in range(len(cbg)):
            for z in range(len(cbg)):
            # Populate the 'odw' array with the indices and values from the 'cbg_OD'
    matrix
                odw[k * len(cbg) + z][0] = k + 1
                odw[k * len(cbg) + z][1] = z + 1
                odw[k * len(cbg) + z][2] = cbg_OD[k][z] #this loop populates the odw
    array with values. The first and second columns of odw
                                                    #store the indices (i+1 and j+1) of
    the loops, and the third column stores
                                                    #the values from the cbg_OD array at
    indices [i][j].



    # Print the 'odw' array
        #print(odw)  # This is a NumPy array.

    # Convert the 'odw' array to a pandas DataFrame 'd1' with column names 'O', 'D', and
    'weight'
        d1 = pd.DataFrame(odw, columns=['O', 'D', 'weight'])
        st_mat = f'Adjacency_Matrix_OD/Static_Com_Matrix_2h/odw_{i+1}.csv'
        d1.to_csv(st_mat, index=False)


    # Create a list of edges from the 'O' and 'D' columns of DataFrame 'd1'
        edges = [(edge[0], edge[1]) for edge in zip(d1['O'], d1['D'])]

    # Display the first few rows of DataFrame 'd1' (only if the 'display' function is
    available)
        #display(d1.head())

    # Print the number of edges in the 'edges' list
        print(len(edges))
        if 'G' in locals() and isinstance(G, nx.Graph):
            G.clear()

        G=nx.DiGraph() #an empty directed graph G created wtih networkx library and
    .digraph() function.The DiGraph() function
                    #initializes a directed graph object that can store nodes and edges
```

```
    with optional attributes.
        for e in range (len(edges)):
            if d1.weight[e] != 0.0:
                G.add_edge(d1.O[e],d1.D[e], weight=d1.weight[e]) #The weight of the edge
    is assigned using the corresponding value from
                                                                #the 'weight' column of d1.

        G.nodes
        print(G)
        #print(G.nodes)
        #print(len(G.nodes))

        import igraph as ig
        from infomap import Infomap
        import matplotlib.pyplot as plt

    # Create an instance of the Infomap algorithm
        im = Infomap(silent=True)

    # Add a networkx graph 'G' to the Infomap instance
        im.add_networkx_graph(G)  # Make sure 'G' is defined before this line

    # Run the Infomap algorithm with specified settings
        im.run(no_self_links=True, flow_model="directed", ftree=True, num_trials=100,
    variable_markov_time=True)

    # Get and print modules at depth level 1
        #print(im.get_modules(depth_level=1))

    # Get and print modules at the finest granularity
        #print(im.get_modules(depth_level=-1))

    # Iterate through the nodes in the Infomap tree
        """
        for node in im.tree:
            if node.is_leaf:
            # Print module ID of each leaf node
                print(node.module_id) """

    # Save results to a DataFrame (if desired)
        df_results = im.get_dataframe()
        df_name = f'Detected_Com_Each_Snapshot_2h/C{i+1}.csv'
        df_results.to_csv(df_name, index=False)
    #!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!1

    # Write the flow tree to a file (if desired)
        ff_name = f'ftree files 2h/C{i+1}.ftree'
        im.write_flow_tree(ff_name) #!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

    # Get the list of modules at depth level 1
        com_1 = list(im.get_modules(depth_level=1).values())
        x=im.add_networkx_graph(G)
    # Get the length of the network (assuming 'x' contains node IDs)
        length = len(x)
        print(length)

    # Initialize an array to store node IDs
        key = np.zeros(length)
        a = 0

    # Fill the 'key' array with node IDs from 'x'
```

```
        for r in range(1, length + 1):
            key[a] = x[r - 1]
            a += 1

    # Create a dictionary mapping node IDs to communities at depth level 1
        dic_1 = dict(zip(key, com_1))

    # Print communities identified at depth level 1
        #print(com_1)

    # Print dictionary mapping node IDs to communities at depth level 1
        #print(dic_1)

    # Create a list of modules at the finest granularity (depth level -1)
        com_2 = list(im.get_modules(depth_level=-1).values())

    # Create a dictionary mapping node IDs to communities at finest granularity
        dic_2 = dict(zip(key, com_2))
        #print(com_2)
        #print(dic_2)

    # Create a DataFrame with 'Key' and 'Value' columns
        col_name = f'Ct{i+1}_lv1'
        df = pd.DataFrame({'Key': list(dic_1.keys()), col_name: list(dic_1.values())})
    #!!!!!!!!!!!!!!!!!!!!

    # Save DataFrame to CSV file (uncomment to save)
        csv_name = f'Actual_Nodes_Communities_2h/C{i+1}.csv'
        df.to_csv(csv_name, index=False) #!!!!!!!!!!!!!!!!!!!!!!!!!!!!!11

        #print(df)

        df.index += 1
        cbg_new = cbg_new.sort_values('objectid_1')
        if 'Key' in cbg_new.columns:
            cbg_new.drop(columns=['Key'], inplace=True)
        cbg_new=pd.merge(cbg_new, df, left_on='objectid_1', right_on='Key',how='left')
        #print(cbg_new)

    # Map communities from level 1 to the DataFrame
    #cbg_new['com_lv1'] = cbg_new.index.map(dic_1)

    # Map communities from the finest granularity to the DataFrame
    #cbg_new['com_lvm1'] = cbg_new.index.map(dic_2)

    # Plot using 'com_lv1' as the color code
        #cbg_new.plot('Ct21_lv1', figsize=(15, 15), legend=True, cmap='GnBu')
    #!!!!!!!!!!!!!!!!!!!

    # Print the plot using 'com_lvm1' as the color code
    #print(cbg_new.plot('Ct1_lvm1', figsize=(15, 15), legend=True, cmap='GnBu'))
    print(cbg_new)
```

In [ ]:
```
print(cbg_new)
#cbg_new.to_file('Com_Shapefiles/GT2h_Static_Communities.shp')
```

Step 2: Community Membership Graph Construction

Community Membership Graphs are generated from each snapshot by basically adding edges to graphs if two endpoints within the same community. Community Membership Graphs are stored in CCMG dictionary.

In [ ]:
```
# Define the path where the community assignment CSV files are located
path1 = "Actual_Nodes_Communities_2h/"

# Get a list of all CSV files in the specified path

all_files = sorted(glob.glob(path1 + "*.csv"), key=os.path.getmtime)
print(all_files)
print(len(all_files))  # Print the number of CSV files found

# Dictionary to store Community Membership Graphs (CCMGs)
CCMG = {}

# Loop through each CSV file
for i in range(len(all_files)):
    # Read the CSV file and create a DataFrame
    dt = pd.read_csv(all_files[i], delimiter=',')
    #col_name = f'Ct{i+1}_lv1'
    df_cm = pd.DataFrame(dt)
    print(df_cm)

    # Create an empty Graph for Community Membership (CMG)
    G_CMG = nx.Graph()

    # Add nodes from the 'Key' column of the DataFrame
    Nodes = [node for node in df_cm['Key']]
    G_CMG.add_nodes_from(Nodes)

    # Initialize the list to store pairs of nodes in the same community
    D_cmg = []

    # Loop through all pairs of nodes in the DataFrame
    for k in range(len(df_cm)):
        for j in range(k+1, len(df_cm)):
            # Check if nodes are not the same and belong to the same community
            if k != j and df_cm[f'Ct{i+1}_lv1'][k] == df_cm[f'Ct{i+1}_lv1'][j]:
                G_CMG.add_edge(df_cm.Key[k], df_cm.Key[j])  # Store the pair in the
CMG

    # Add edges to the CMG based on community membership
    #G_CMG.add_edges_from(D_cmg)

    # Create a name for the CMG and store it in the CCMG dictionary
    graph_name = f'G_CMG{i+1}'
    CCMG[graph_name] = G_CMG


print(CCMG)
print(len(CCMG))# Print the dictionary of Community Membership Graphs

# Plot the Community Membership Graph
plt.figure(figsize=(16, 16))
ax = plt.gca()
pos = nx.random_layout(CCMG['G_CMG1'])

nx.draw_networkx(CCMG['G_CMG1'], pos, ax=ax, width=0.1)
# Display the plot
plt.show()

# Find cliques (complete subgraphs) in the Community Membership Graph
y = nx.find_cliques(CCMG['G_CMG1'])
```

```
display(y)
print(len(list(y)))  # Print the list of cliques
```

Step 3: Consensus Graph "GC" Generation

Consensus graph constructed based on community membership graphs. Each CMG and their nodes are iterated and, if an edge occured within a CMG, it is added to consensus graph.

Empty Consensus Matrix "CM" --> Iterate over CMG --> Update CM if an edge exist in CMG (by adding "1" to corresponding Aij element of CM --> Generate GC from adjacency matrix CM.

Here, maximum value of an element of CM can be 12 since we have 12 snapshots. This process allow us to consider how strongly an edge is connected troughout the time.

In [ ]:
```
# Create an array to store the consensus matrix (CM)
CM = np.zeros((len(cbg), len(cbg)))

# Loop through each Community Membership Graph (CMG)
for i in range(len(CCMG)):
    # Convert the CMG into an edge list DataFrame
    el = nx.to_edgelist(CCMG[f'G_CMG{i+1}'])
    el1 = pd.DataFrame(el, columns=['O', 'D', 'W'])
    print(len(el1))

    # Update the consensus matrix using the edge list information
    for j in range(len(el1)):
        u = el1.O[j]
        v = el1.D[j]
        CM[int(u - 1)][int(v - 1)] = CM[int(u - 1)][int(v - 1)] + 1

# Display the full consensus matrix
pd.set_option('display.max_columns', None)
display(CM)
np.savetxt('Adjacency_Matrix_OD/Adj_Mx_Consensus_Network_12.csv', CM, delimiter=',')

# Create an array to store the consensus matrix in list format
#CM1 = np.zeros((len(cbg) * len(cbg), 3))
CM1 = []
"""for i in range(len(cbg)):
    for j in range(len(cbg)):
        CM1[i * len(cbg) + j][0] = i + 1
        CM1[i * len(cbg) + j][1] = j + 1
        CM1[i * len(cbg) + j][2] = CM[i][j]"""

for i in range(len(cbg)):
    for j in range(i + 1, len(cbg)):  # Iterate only for j > i to avoid duplicating
work
        CM1.append([i + 1, j + 1, (CM[i][j] + CM[j][i])])



print(CM1)
np.savetxt('Adjacency_Matrix_OD/Adj_Mx_Consensus_Network_12_List.csv', CM1,
delimiter=',')
# Create a DataFrame from the consensus matrix in list format
dcm = pd.DataFrame(CM1, columns=['O', 'D', 'weight'])
d_edges = [edge for edge in zip(dcm['O'], dcm['D'])]
print(dcm)
```

```
# Create a new Graph for the consensus network
GC = nx.Graph()

# Add edges with weights to the consensus network
for i in range(len(d_edges)):
    if dcm.weight[i] != 0.0:
        GC.add_edge(dcm.O[i], dcm.D[i], weight=dcm.weight[i]/12)

# Display nodes in the consensus network
print(GC.nodes)

# Plot the consensus network
"""plt.figure(figsize=(16, 16))
ax = plt.gca()
pos = nx.random_layout(GC)

nx.draw_networkx(GC, pos, ax=ax, width=0.1)"""
# Display the plot
plt.show()
# Display edges in the consensus network
print(len(GC.edges))

# Display the number of nodes in the consensus network
print(len(GC.nodes))
```

In [ ]:
```
plt.figure(figsize=(16, 16))
ax = plt.gca()
pos = nx.random_layout(GC)

nx.draw_networkx(GC, pos, ax=ax, width=0.1)
#Display the plot
plt.show()
```

In [ ]:
```
fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.matshow(CM)
fig.colorbar(cax)

#plt.matshow(CM)
plt.show()
print(el1)
```

Step 4: Consensus Communities GC_Com

Here we have applied infomap algorithm to GC consensus graph. Result is the consensus communities.

In [ ]:
```
im1 = Infomap(silent=True)
im1.add_networkx_graph(GC)
im1.run(flow_model="undirected", ftree=True, num_trials = 100)
#print(im1.get_modules(depth_level=1))  preferred_number_of_modules=5,
weight_threshold=0.1, teleportation_probability regularized=True
#print(im1.get_modules(depth_level=2))  entropy_corrected=True, markov_time=5,
num_trials=100, regularized=True, inner_parallelization=True
#entropy_corrected=True, preferred_number_of_modules=5, weight_threshold=0.9,
teleportation_probability=0.05, regularized=True)

for node in im1.tree:
    if node.is_leaf:
        print(node.module_id)
```

```
df_results1=im1.get_dataframe()
df_results1.to_csv("Detected_Com_Each_Snapshot_2h/GC_Communities_2h.csv")
im1.write_flow_tree("ftree files 2h/GC_2h.ftree")
print(df_results1)

glen=im1.add_networkx_graph(GC)
print(glen)
com2_1=list(im1.get_modules(depth_level=1).values())
length=len(glen)
print(length)
key=np.zeros(length)
a=0
for i in range(1,length+1):
    key[a]=glen[i-1]
    a += 1
dic2_1 = dict(zip(key, com2_1))
#print(com2_1) #communities that are identified
#print(dic2_1) #communities assigned to each district


com2_2=list(im1.get_modules(depth_level=2).values())
dic2_2 = dict(zip(key, com2_2))
#print(com2_2)
#print(dic2_2)

com2_3=list(im1.get_modules(depth_level=3).values())
dic2_3 = dict(zip(key, com2_3))
#print(com2_3)
#print(dic2_3)

com2_4=list(im1.get_modules(depth_level=4).values())
dic2_4 = dict(zip(key, com2_4))
#print(com2_4)
#print(dic2_4)


GC_Com = pd.DataFrame({'Key': list(dic2_1.keys()), 'coms': list(dic2_1.values())})
#'Value2': list(dic2_2.values()), 'Value3': list(dic2_3.values()), 'Value4':
list(dic2_4.values())
# Save DataFrame to CSV file
GC_Com.to_csv('Actual_Nodes_Communities_2h/Consensus_Com_2h/GC_Com_2h.csv',
index=False)
print(GC_Com)
```

In [ ]:
```
print(GC_Com)
```

In [ ]:
```
GC_Com.index += 1
if 'Key' in cbg_new.columns:
        cbg_new.drop(columns=['Key'], inplace=True)
cbg_new=pd.merge(cbg_new, GC_Com, left_on='objectid_1', right_on='Key',how='left')
print(cbg_new)

print(cbg_new.plot('coms', figsize=(15, 15),legend=True,cmap='GnBu'))
#print(cbg_new.plot('Value2_x', figsize=(15, 15),legend=True,cmap='GnBu'))
#print(cbg_new.plot('value3', figsize=(15, 15),legend=True,cmap='GnBu'))
#print(cbg_new.plot('value4', figsize=(15, 15),legend=True,cmap='GnBu'))
```

In [ ]:
```
cbg_new.to_file('Com_Shapefiles/GC_Community_2h.shp')
```

Step 5: Dynamic Graph Generation

In [ ]:
```
# Print the DataFrame with community assignments
print(len(GC_Com))

# Dictionary to store dynamic communities
Dynamic_Com = {}

# Loop through each unique community at depth level 1
for i in range(len(set(GC_Com['coms']))):
    # Create a name for the dynamic community
    g_name = f'GD{i+1}'

    # Create a Graph for the dynamic community
    GD = nx.Graph()

    # Loop through the GC_Com DataFrame to identify nodes belonging to the dynamic
community
    for j in range(1, len(GC_Com) +1):
        if GC_Com.coms[j] == i+1:
            node = GC_Com.Key[j]
            if node not in GD:
                GD.add_node(node)

    # Store the dynamic community Graph in the dictionary
    Dynamic_Com[g_name] = GD

# Print the dictionary of dynamic communities
print(Dynamic_Com)

# Print nodes in the first dynamic community
print(Dynamic_Com['GD1'].nodes)

# Print nodes in the second dynamic community
#print(Dynamic_Com['GD2'].nodes)

# Print nodes in the third dynamic community
```

Step 5.2: Edge Collection for each snapshot

Dynamic_Matrix is a dictionary to store edges of each snapshot

In [ ]:
```
path2 = "Adjacency_Matrix_OD/Static_Com_Matrix/"
matrix_files = sorted(glob.glob(path2 + "*.csv"), key=os.path.getmtime)
print(matrix_files)
print(len(matrix_files))

Dynamic_Matrix = {}


for k in range(len(matrix_files)):
    data1 = pd.read_csv(matrix_files[k], delimiter = ',')
    DM1 = pd.DataFrame(data=data1)
    Dynamic_Matrix[f'DM_{k+1}'] = DM1


print(Dynamic_Matrix)
```

Step 5.3: Adding Edges to dynamic graphs

This part automates the edge adding process to dynamic graphs.

The dynamic graphs with nodes and no edges filled with edges from corresponding snapshot.

the code first calls first dynamic graph, GD1, and bike trips from first time interval G1. it iterates over the edges within G1, checks if two endpoints of an edge (i,j) exist with in GD1, has a wieght >= 5 and, i is not equal to j. If the criterias are met, then the edge is added to the graph. Resulting gynamic graph saved in GDi_t, time evolving dynamic graph dictionary, under the name of GD1_1. The first number indicates label of consensus community/dynamic graph and second number indicates label of time interval.

GD1_1,.., GD1_12, ..., GD1_48 . . . GD6_1, ..., GD6_12, ..., GD6_48

In [ ]:
```
# Convert the arrays in Dynamic_Matrix dictionary to DataFrames
for i in range(len(Dynamic_Matrix)):
    Dynamic_Matrix[f'DM_{i+1}'] = pd.DataFrame(Dynamic_Matrix[f'DM_{i+1}'],
columns=['O', 'D', 'weight'])
print(Dynamic_Matrix)

# Dictionary to store time-evolving dynamic graphs
GDi_t = {}

# Loop through each dynamic community
for z in range(len(Dynamic_Com)):
    # Loop through each dynamic adjacency matrix
    for k in range(len(Dynamic_Matrix)):
        dmx = Dynamic_Matrix[f'DM_{k+1}']
        dcx = nx.Graph()

        # Add nodes from the dynamic community graph to the evolving graph
        dcx.add_nodes_from(Dynamic_Com[f'GD{z+1}'])

        # Loop through each row in the dynamic adjacency matrix
        for j in range(len(dmx)):
            # Check if both nodes are in the evolving graph and have non-zero weight
            if dmx.O[j] in dcx.nodes and dmx.D[j] in dcx.nodes and dmx.weight[j] >=
5.0 and dmx.O[j] != dmx.D[j]:
                dcx.add_edge(dmx.O[j], dmx.D[j], weight=dmx.weight[j])
        print(dcx)
        # Create a name for the evolving graph and store it in the dictionary
        graph_name = f'GD{z+1}_{k+1}'
        GDi_t[graph_name] = dcx

# Print the dictionary of time-evolving dynamic graphs
print(GDi_t)
```

Step 6: Sub Community Detection

In this part, community detection is applied to all dynamic graphs to detect sub communities.

SCD_dict is a dictionary to save all com. det. results

this part iterate over dynamic graphs from GDi_t and implement com. det. to them.

additionally, the code generates single node assignments. For example, while generating 6-7 connected communities, it also gives communities with single nodes which doesnt seem very nice. therefore i hale adopt a part where these single node communities converted into 0 to be removed afterward.

In [ ]:
```
# DataFrame to store subcommunity assignments for each snapshot
SCD_dict = {}
```

```
for z in range(len(Dynamic_Com)):
    SCD_t = pd.DataFrame()

# Loop through each dynamic community subgraph in GD1_t
    for i in range(0, 48):
    # Create an Infomap instance for the current dynamic community subgraph
        im2 = 0
        im2 = Infomap(silent=True)
        im2.add_networkx_graph(GDi_t[f'GD{z+1}_{i+1}'])
        im2.run(no_self_links=True, flow_model="undirected", ftree=True,
num_trials=100)

    # Generate the flow tree file for visualization
        ffilename = f"ftree files 2h/SCD{z+1}_2h_{i+1}.ftree"
        im2.write_flow_tree(ffilename)

    # Calculate the length of the nodes in the current graph
        glen1 = im2.add_networkx_graph(GDi_t[f'GD{z+1}_{i+1}'])
        length1 = len(glen1)

    # Initialize arrays for keys and communities
        key1 = np.zeros(length1)
        a = 0
        for j in range(1, length1 + 1):
            key1[a] = glen1[j - 1]
            a += 1

    # Get communities at depth level 1
        com = list(im2.get_modules(depth_level=1).values())
        dic = dict(zip(key1, com))

        value_counts = {}
        for value in dic.values():
            value_counts[value] = value_counts.get(value, 0) + 1
        for key, value in dic.items():
            if value_counts[value] == 1:
                dic[key] = 0

    # Create a new dictionary with only the key-value pairs where the value appears
more than once
        #SCDi_t = {key: value for key, value in dic.items() if value_counts[value] >
1}

    # Create a DataFrame with subcommunity assignments for the current snapshot
        SCDi_t = pd.DataFrame({'Key': list(dic.keys()), f't{i+1}':
list(dic.values())})

    # Update the main DataFrame for subcommunity assignments
        if SCD_t.empty:
            SCD_t = SCDi_t
        else:
            SCD_t = pd.merge(SCD_t, SCDi_t, on='Key', how='outer')

# Sort and reset index of the DataFrame
    SCD_t.sort_values('Key', inplace=True)
    SCD_t.reset_index(drop=True, inplace=True)
    SCD_dict[f'SCD{z+1}_t'] = SCD_t

# Print the final DataFrame with subcommunity assignments
```

```
        print(SCD_t)

    # Save the DataFrame to a CSV file
        SCD_t.to_csv("Dynamic_Subcommunities_2h/SCD{z+1}_t_Communities.csv")
```

This part is for saving the results. be careful on the SCD_i.index part

In [ ]:
```
for i in range(len(SCD_dict)):
    column_name = f'SCD{i+1}_t'

    # Check if the column exists in SCD_dict before attempting to access it
    if column_name in SCD_dict:
        SCD_i = SCD_dict[column_name]
        SCD_i.index += 1 ######## be careful!! do not run this specific line twice,
otherwise indexes are shifted and result is distorted.
        if 'Key' in cbg_new.columns:
            cbg_new.drop(columns=['Key'], inplace=True)

        # Merge with suffixes to avoid duplicate columns
        cbg_new = pd.merge(cbg_new, SCD_i, left_on='objectid_1', right_on='Key',
how='left', suffixes=('', f'_{i+1}'))

    print(cbg_new)
```

In [ ]:
```
print(cbg_new)
```

In [ ]:
```
print(cbg_new.plot('t48', figsize=(15, 15),legend=True,cmap='GnBu'))
```

In [ ]:
```
num_rows = 6
num_cols = 8

# Create a figure and an array of subplots
fig, axes = plt.subplots(num_rows, num_cols, figsize=(15, 15))
axes = axes.flatten()
for i in range(0, 48):
    ax = axes[i]
    column = f't{i+1}'
    cbg_new.plot(column, ax=ax, legend=True, cmap='GnBu')
    ax.set_title(column)  # Set the title of the subplot to the column name
    ax.set_axis_off()     # Turn off axis labels and ticks

# If there are more axes than columns to visualize, hide the extra subplots
#for j in range(48, len(axes)):
    #axes[j].axis('off')

# Adjust layout to prevent overlap
plt.tight_layout()
plt.savefig('output_plot2.png', dpi=600)
plt.show()
```

In [ ]:
```
cbg_new.to_file('Com_Shapefiles/GD_Coms4.shp')
```

In [ ]:
```
print(cbg_new)
```

In [ ]:

Appendix B – Supplementary Material

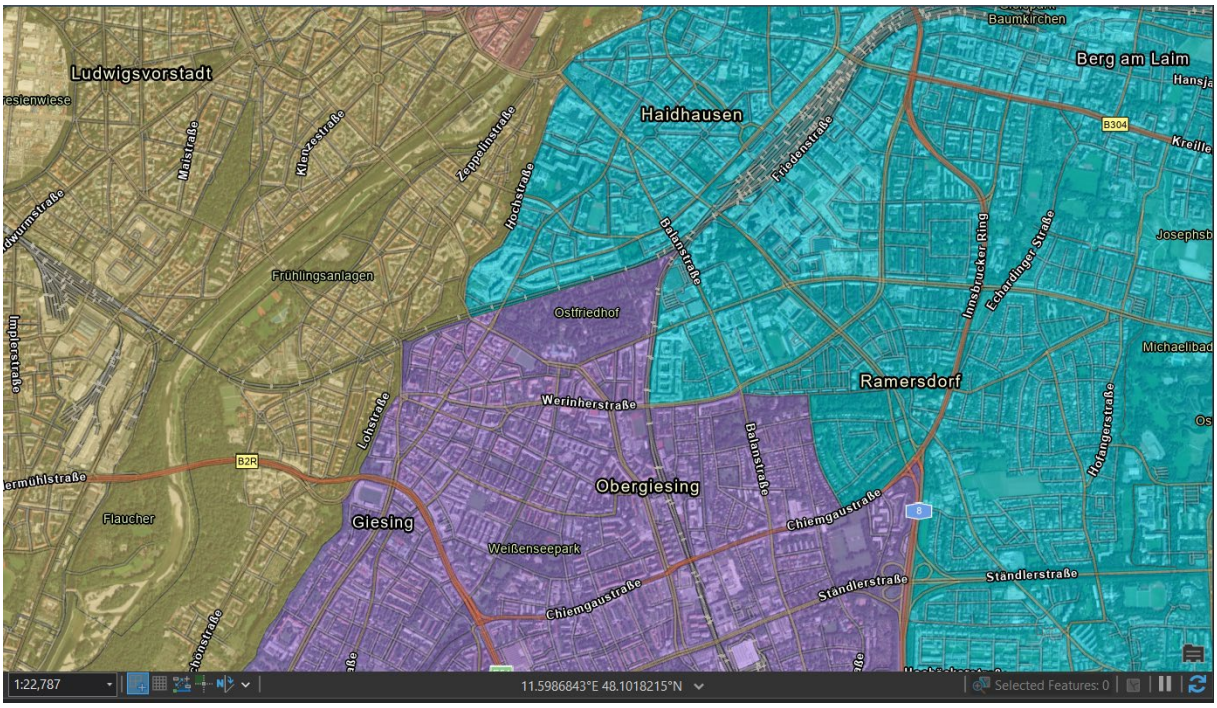


Figure 20. Borders of Consensus Communities 2, 4, 6.

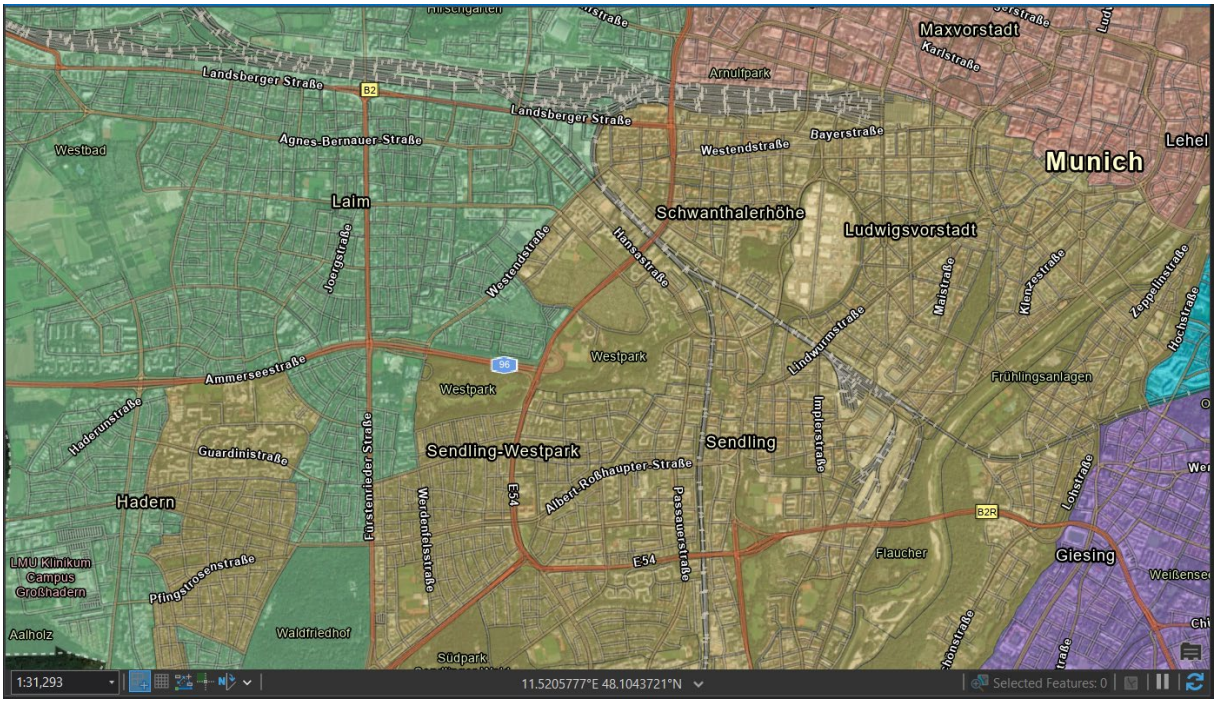


Figure 21. Borders of Consensus Communities 1, 2, 3, 6.

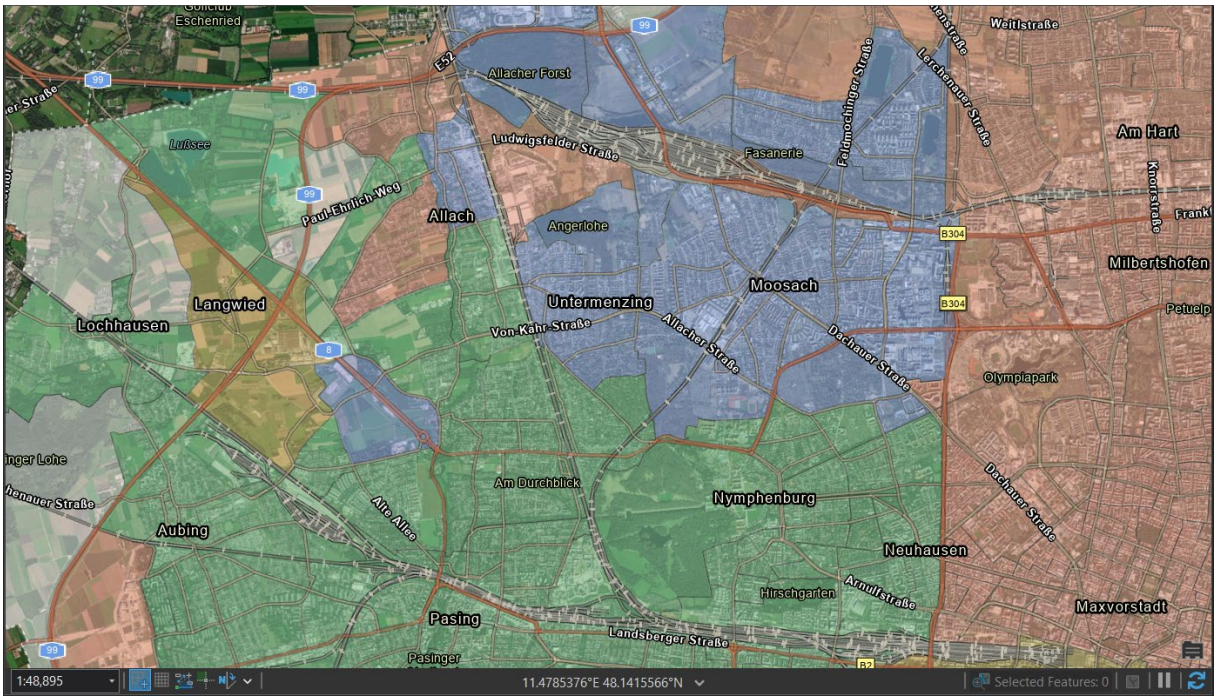


Figure 22. Borders of Consensus Communities 1, 3, 5.

Figure 23. Static Community Detection Results for 12 intervals. (T1 = 00:00 - 02:00)

Figure 24. Dynamic Sub-Community Detection Results on 48 intervals. (T14 = 06:30 – 07:00)