FEDERATED LEARNING ASSISTED FRAMEWORK TO PERIODICALLY IDENTIFY USER COMMUNITIES IN URBAN SPACE

Cláudio Diego Teixeira de Souza

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia de Sistemas e Computação.

Orientadores: José Ferreira de Rezende
Carlos Alberto Vieira Campos

Rio de Janeiro
Abril de 2025

FEDERATED LEARNING ASSISTED FRAMEWORK TO PERIODICALLY IDENTIFY USER COMMUNITIES IN URBAN SPACE

Cláudio Diego Teixeira de Souza

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Orientadores: José Ferreira de Rezende
Carlos Alberto Vieira Campos

Aprovada por: Prof. Diego Leonel Cadette Dutra
Prof. Miguel Elias Mitre Campista
Prof. Aline Marins Paes Carvalho

RIO DE JANEIRO, RJ – BRASIL
ABRIL DE 2025

Teixeira de Souza, Cláudio Diego

Federated Learning assisted framework to periodically identify user communities in urban space/Cláudio Diego Teixeira de Souza. – Rio de Janeiro: UFRJ/COPPE, 2025.

XV, 105 p.: il.; 29, 7cm.

Orientadores: José Ferreira de Rezende

Carlos Alberto Vieira Campos

Tese (doutorado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2025.

Referências Bibliográficas: p. 94 – 105.

1. Community identification. 2. Federated Learning. 3. Spatio-temporal mobility. 4. Urban mobility. I. de Rezende, José Ferreira *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

FEDERATED LEARNING ASSISTED FRAMEWORK TO PERIODICALLY IDENTIFY USER COMMUNITIES IN URBAN SPACE

Cláudio Diego Teixeira de Souza

Abril/2025

Orientadores: José Ferreira de Rezende
Carlos Alberto Vieira Campos

Programa: Engenharia de Sistemas e Computação

A identificação de indivíduos com comportamentos e padrões de mobilidade semelhantes tornou-se essencial para melhorar o funcionamento dos serviços urbanos. Entretanto, como esses padrões podem variar com o tempo, essa identificação precisa ser feita periodicamente. Além disso, uma vez que os dados de mobilidade expressam a rotina dos indivíduos, a privacidade deve ser garantida. Neste trabalho, propomos um *framework* para detectar e agrupar periodicamente indivíduos com semelhanças de comportamento em comunidades. Para isso, criamos um modelo *autoencoder* para extrair *features* de mobilidade espaço-temporal de dados brutos do usuário em intervalos periódicos. Usamos a Federated Learning (FL) como abordagem de treinamento para preservar a privacidade e aliviar os custos de treinamento e comunicação. Para determinar o número de comunidades sem arriscar um valor arbitrário, comparamos as escolhas de dois métodos probabilísticos, o Critério de Informação de Akaike (AIC) e o Critério de Informação Bayesiano (BIC). Como as comunidades são atualizadas periodicamente, também analisamos o impacto de amostras antigas no *framework* proposto. Por fim, comparamos o desempenho de nossa solução baseada em FL com uma abordagem de treinamento centralizado. Analisamos as métricas de similaridade e dissimilaridade em amostras de mobilidade e o tempo de contato dos indivíduos em três cenários diferentes. Nossos resultados indicam que a AIC supera a BIC ao escolher o número de comunidades, embora ambas satisfaçam as métricas de avaliação. Também descobrimos que o uso de amostras mais antigas beneficia cenários espaço-temporais mais complexos. Por fim, não foram detectadas perdas significativas quando comparadas a uma abordagem de treinamento centralizado, reforçando as vantagens de usar o método baseado em FL.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

FEDERATED LEARNING ASSISTED FRAMEWORK TO PERIODICALLY IDENTIFY USER COMMUNITIES IN URBAN SPACE

Cláudio Diego Teixeira de Souza

April/2025

Advisors: José Ferreira de Rezende
Carlos Alberto Vieira Campos

Department: Systems Engineering and Computer Science

Identifying individuals with similar behaviors and mobility patterns has become essential to improving the functioning of urban services. However, since these patterns can vary over time, such identification needs to be done periodically. Furthermore, once mobility data expresses the routine of individuals, privacy must be guaranteed. In this work, we propose a framework for periodically detecting and grouping individuals with behavioral similarities into communities. To accomplish this, we built an autoencoder model to extract spatio-temporal mobility features from raw user data at periodic intervals. We used Federated Learning (FL) as a training approach to preserve privacy and alleviate time-consuming training and communication costs. To determine the number of communities without risking an arbitrary number, we compared the choices of two probabilistic methods, the Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC). Since the communities are updated periodically, we also analyzed the impact of aged samples on the proposed framework. Finally, we compared the performance of our FL-based solution to a centralized training approach. We analyzed similarity and dissimilarity metrics on mobility samples and the contact time of individuals in three different scenarios. Our results indicate that AIC outperforms BIC when choosing the number of communities, although both satisfy the evaluation metrics. We also found that using older samples benefits more complex spatio-temporal scenarios. Finally, no significant losses were detected when compared to a centralized training approach, reinforcing the advantages of using the FL-based method.

Contents

List of Figures

List of Tables

List of Abbreviations

ACC	Accumulative Window, p. 29
AE	Autoencoder, p. 11
AF	Activation Function, p. 11
AIC	Akaike Information Criterion, p. 4
ARI	Adjusted Rand Index, p. 41
BIC	Bayesian Information Criterion, p. 4
CAE	Convolutional Autoencoder, p. 11
DL	Decoding Layer, p. 11
DTN	Delay-tolerant network, p. 64
EL	Encoding Layer, p. 11
FCAE	Full Convolutional Autoencoder, p. 3
FL	Federated Learning, p. 3
FTL	Federated Transfer Learning, p. 16
GMM	Gaussian Mixture Model, p. 21
GPS	Global Positioning System, p. 24
HFL	Horizontal Federated Learning, p. 16
IFPCU	Intra-Flooding based on Periodic Community Update, p. 70
LS	Latent Space, p. 11
MANET	Mobile ad hoc network, p. 65
MBC	Model-Based Clustering, p. 19

MSE	Mean Square Error, p. 41
OMSN	Opportunistic Mobile Social Network, p. 4
ONE	Opportunistic Network Environment, p. 70
PCU	Periodic Community Update, p. 67
SLI	Sliding Window, p. 29
SMC	Secure Multi-party Computation, p. 17
SSIM	Structural SIMilarity, p. 41
TTL	Time To Live, p. 70
VAE	Variational Autoencoder, p. 11
VFL	Vertical Federated Learning, p. 16

Chapter 1

Introduction

As the process of urban development advances and the demands for improvements in people's quality of life grow in aspects such as education, health and transportation, for example, it becomes important to comprehend the urban routines across various domains. Understanding how these dynamics occur is a fundamental part of creating urban spaces focused on collective needs and capable of providing benefits to the citizens' well-being, sustainability and smart development [1, 2]. In this context, the analysis of urban mobility, i.e., the presence of people in certain spaces and their flows and interactions, is capable of revealing valuable information about both individual and collective spatio-temporal behavior patterns, such as commuting demands [3], stopover areas [4], points of interest [5], etc.

Identifying these behavioral patterns helps both to understand the relationship between mobility and the use of resources – as well as their allocation – in the social and spatial spheres of cities [6], and to improve some strategic sectors, such as transportation systems [7], epidemic modeling [8], socioeconomic indicators [9], among others. Since different individuals (or users) can have similar behavioral patterns, detecting those with high behavioral similarity would allow us to form what the literature calls communities [10].

Community detection has proved valuable in various fields of study, from computer science to sociology and biology [11], and consists of dividing a network into subgroups (the communities) in which the nodes within a subgroup are strongly connected, while nodes in different subgroups have weaker connections. From this, individuals that move within the urban space can be considered vertices of a network, and the similarities between their behaviors are the edges that connect them. To identify these similarities, the behavior of each user can be characterized by their spatio-temporal displacement patterns, thus allowing the detection of user communities in urban spaces.

1.1 The problem of identifying user communities in urban spaces

As these communities are subject to the mobility characteristics of users, which can vary over time, they are not perennial entities but can have their compositions changed according to the behavior of such individuals as time progresses [12]. These changes can lead to valuable inferences about how people interact, for example, during the COVID-19 pandemic, when such interactions were one of the crucial factors in the disease's spread [13]. As a result, both the composition and the periodic evolution of communities become relevant issues. Therefore, setting the appropriate number of communities is important so that each group is composed only of users with the most similar behaviors. In turn, periodic evolution implies that at each subsequent time interval in which there is a significant change in individuals' behavior, the set of communities must be adjusted.

Although the topic of community detection has been widely studied, few works have focused on the periodic evolution of communities; some of them include [14–17]. Furthermore, even fewer studies address this issue when applied to the urban environment based on spatio-temporal user mobility information [18]. In parallel, most of the studies that relate community detection to human mobility do not directly address the use of spatio-temporal behavioral patterns for grouping individuals. Instead, they focus on communities based on urban structures [19–21], where certain regions are grouped together based on implicit connections, resulting from the origins and destinations of individual trajectories. However, analyzing individuals' journeys can reveal important information about collective behavior as users move through urban space. Therefore, a few other efforts have been dedicated to detecting these behaviors and grouping individuals.

In particular, [22] proposes a framework based on deep learning and analyzes the performance of different autoencoder structures to learn spatio-temporal features from user mobility data and group them according to their similarities. By using these structures to extract non-linear spatio-temporal features from image-based mobility representations, the work improves community identification compared to other existing methods. Yet, like other efforts, this work, although presenting relevant findings, does not address the temporal evolution mentioned above. In addition, a common strategy in existing works, including this one, refers to centralized data processing, i.e., using users' mobility data sets as if they were all available centrally on a server. However, a current discussion refers to the impracticality of gathering in a data center large amounts of data, usually produced by a massive number of users, due to possible problems in communication networks and because this sharing may be subject to privacy restrictions and data sovereignty laws [23].

1.2 Objectives

Therefore, the aim of this work is to detect communities of users in the urban space, considering both the need to periodically adjust the set of communities and the impracticality of centralizing the mobility data of these individuals. To overcome the problem of data centralization, we propose the adoption of Federated Learning (FL). [24].

Thus, this work aims more specifically to:

1. Conduct a literature review on community detection using machine learning models and their training approaches.

2. Analyze the strategies adopted in these works to ensure data privacy and the periodic evolution of communities.

3. Develop a FL-based framework for periodic community detection that meets data privacy requirements and regularly adjusts the number and composition of communities.

4. Evaluate the impact of probabilistic methods for choosing the number of communities in each interval.

5. Compare the influence of using aged mobility samples and only the most recent ones in training the proposed model.

6. Test and evaluate the effectiveness of the proposed model through experiments with different real and synthetic mobility data sets, analyzing similarity and dissimilarity metrics between community users.

7. Compare the results obtained with those of a traditional (centralized) training approach.

8. Evaluate the quality of communities in a case study.

In the proposed framework, image-based representations of user mobility are created periodically and processed by a Full Convolutional Autoencoder (FCAE) model, so that latent spatio-temporal features are extracted from these representations. From these features, it is possible to identify similarities among users' geographical preferences, enabling their grouping into communities.

As for the FL, it has been adopted in different contexts to provide a privacy-preserving mechanism, operating on decentralized data and preventing it from traveling over the network. It is characterized by collaborative learning, in which multiple devices at the edge of the network contribute to training a shared model while maintaining all training data locally [25]. In addition, FL is based on the

data minimization principle, so that minimal updates about the training task are transmitted to users, who can process the data as soon as it is released, while an aggregator entity retains minimal information, discarding it as quickly as possible [26]. In this way, FL solves fundamental problems related to user data privacy during model training, whereas it can contribute to the periodic analysis of mobility data for user community detection.

To avoid predetermining an arbitrary number of communities, we propose a choice based on a probabilistic method at each update. So, we compared the choices made by the Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC). In this way, one does not need to previously infer the number of communities, i.e., before the data is analyzed. Finally, since the communities are updated periodically, we also assessed the influence of aged samples on both model learning and community detection. Therefore, we propose two window-based strategies for the processing of samples in local training. Each strategy defines a number of historical samples to be used to train the FCAE model.

1.3 Research questions and contributions

Through this work, we aim to address three research questions: firstly, based on the premise that the number of communities must vary over time in order for individuals to be better grouped, we want to know the implications of using the probabilistic methods AIC and BIC for this task. Then, since we would like to assess the use of aged samples, the next question concerns the impact of window-based strategies on the model training and the communities detected. Finally, since FL consists of a distributed learning mechanism, we would also like to know whether there is a significant disparity between the results obtained from a traditional centralized training approach and our FL-based solution. To validate our work, we used three publicly available mobility trace sets and analyzed similarity and dissimilarity metrics among mobility samples and average contact time among intra- and inter-community users. We also conducted a case study focused on message forwarding in Opportunistic Mobile Social Networks (OMSN) using community information obtained with our framework in two real mobility scenarios and two synthetic mobility scenarios.

To the best of our knowledge, this is the first work to propose periodic detection of user communities in urban space assisted by FL using spatio-temporal behavioral patterns. It is also the first work to compare different configurations of user communities, resulting from both the size of the training sample set and the probabilistic choice of the number of communities by the AIC and BIC methods. As such, the contributions of this work were summarized as follows:

1. We present an FL-assisted framework for the periodic detection of user communities in urban spaces. The framework allows the number and composition of communities to vary over time.

2. We show that there is no significant disparity between the results of the centralized and FL-based approaches, so it is favorable to use a distributed learning mechanism such as FL to detect user communities in urban spaces, especially for privacy concerns.

3. Based on our framework, which incorporates a scheme for periodic sample production and processing, we confirm that time influences the composition of communities due to changes in the behavior of individuals.

4. We show that, as time progresses, the different choices of the probabilistic method for the number of communities satisfactorily group individuals who have behavioral similarities.

5. We show that our framework produces good results for all the evaluation metrics. In general, these results represent a considerable increase in the average contact time between intra-community individuals.

6. Through our case study, we confirm the quality of the communities detected by the proposed framework and how they can help in the delivery of messages in OMSN, which may motivate their use in other applications.

The remainder of this work is organized as follows: Chapter 2 presents some related works and the preliminary concepts for our proposal. Our proposed solution is described in Chapter 3. In Chapter 4, the experimentation datasets, scenarios, model configuration parameters and a description of the evaluation metrics are presented. In Chapter 5, the obtained results are analyzed. Chapter 6 presents our case study, describing the concept of OMSN, our experimental methodology, and the results obtained. Finally, Chapter 7 presents our conclusions and some directions for future work.

Chapter 2

Related work

In this chapter, we review some works in the literature related to community detection in urban spaces, as well as introduce the main concepts related to autoencoders and Federated Learning, which were used to build our solution.

2.1 Community Detection

Human mobility has been widely investigated in different research areas to portray different aspects of daily life. Given the abundance and proliferation of devices that allow mobility data to be collected, a growing volume of records has driven solutions that seek to extract information about the individuals' behavioral patterns. In this context, a branch of investigation consists of identifying meaningful substructures hidden in these individuals' interactions and behaviors, the so-called communities.

Many definitions and approaches have been proposed for this interdisciplinary problem, which has been treated through different methods over the years [27, 28]. Regarding the communities' detection in urban spaces, two main definitions stand out: grouping physical spaces that share characteristics of public interest [29] and grouping individuals who share behavioral similarities, such as movement patterns or geographical preferences [30].

The first case, focused on connecting the spatial structures in cities, assumes that these connections can be established by the flow of people, goods, money and information [31]. It is mainly related to the transformation of urban spaces into a complex polycentric agglomeration of areas that are interconnected by dynamics and collective interests. Some works include [3, 32–35], among others. Efforts like these enrich the understanding of urban structures and play an important role in the execution of services such as urban planning and traffic modeling [21, 36], for example. In addition, most of them address the issue of the temporal evolution of community structures, based on the premise that, as they vary, urban mobility patterns have implications for the groups identified. On the other hand, they are mainly concerned

with the trajectories' origins and destinations as a way to build a network of urban space structures that are interconnected by the individuals' interests.

To capture aspects that resemble the individuals' behavior, a fine-grained data analysis is necessary. Such analysis can help define trends in the flow of individuals, which, as a result, can improve the provision of services in cities, such as the functioning of intelligent transport systems, epidemic modeling and traffic forecasting [37, 38], as well as adapting connectivity between users and improving communication applications in cluster scenarios[39], for example. Thus, identifying and grouping individuals who share similar mobility characteristics assumes a fundamental role in collective human behavior analysis.

A trend in recent community detection methods is the use of deep learning techniques to identify these structures. In general, this involves learning lower-dimensional vectors from high-dimensional data expressing relationships among units [40, 41]. Many community detection efforts based on deep learning can be found in the literature. Table 2.1 summarizes some of these works, which are analyzed below. For a more complete review of the literature on the use of deep learning for community detection, we suggest that the reader consult [11, 42].

Table 2.1: Summary of works related to detecting communities in urban spaces and their limitations

Based method	Proposal	Strength	Limitation	Community evolution	Data privacy
GNN/GCN	[43–45]	High modularity scores [43] Efficiency [44, 45]	Complexity [43] Fixed number of communities [44], Reliance on known community structures and sufficient labels [45]	Yes [43], No [44, 45]	No [43–45]
CNN	[46–48]	Efficiency in sparse matrices [46], Accuracy improvement [47, 48]	Scalability [46], Complexity and need for large labeled datasets [47], Inaccurate edge recognition [48]	No [46–48]	No [46–48]
Attention NN	[49, 50]	Discriminative and effective embeddings [49], Improved accuracy and flexibility [50]	Need for annotated data [49], Scalability [50]	No [49, 50]	No [49, 50]
Autoencoder	[22] [51, 52]	Intra-community similarity increase [22], Effective dynamic community detection [51], Data privacy protection [52]	Scalability, generalization and privacy [22], Complexity [51], Pre-determined number of communities [52]	Yes [22, 51], No [52]	No [22, 51], Yes [52]
Representation learning	[53, 54]	Improved vertex classification [53], Effective analysis of mobility profiles [54]	Adjustments for different types of networks and complexity [53], Subjective interpretation [54]	No [53, 54]	No [53, 54]

Based on Graph Neural Networks (GNN), [43] proposes a solution for unsupervised community detection. It presents an efficient and balanced approach between performance and speed, capable of achieving modularity scores similar to or higher than those of more complex algorithms, although its own complexity depends on the

number of runs to be adjusted. It also considers the evolution of these communities over time, as it operates with temporal networks, which allows it to reconstruct important longitudinal patterns. Based on Graph Convolutional Networks (GCN), a specific type of GNN, [44] and [45] propose solutions for supervised and unsupervised community detection, respectively. Although [44] performs well on synthetic and real datasets, outperforming traditional approaches such as spectral methods and probabilistic graphical models, the model assumes a fixed number of communities to detect, which limits its application in scenarios where this number varies. In turn, [45] shows superior comparative performance compared to traditional methods. The solution is based only on the structure of the network and the attributes of the nodes. However, it depends on known community structures and an adequate number of labels for effective propagation. All these proposals can be used to detect communities of users in urban spaces, but none of them address data privacy issues, and only [43] considers the evolution of communities.

Focused on efficiency, accuracy and scalability in large-scale and complex social networks, [46] and [47] propose solutions based on Convolutional Neural Networks (CNN) for community detection in Online Social Networks. While the former shows the effectiveness of the convolution technique on sparse matrices and its successful application in real scenarios, the latter shows a significant improvement in the accuracy of community detection compared to traditional approaches. Regarding limitations, [46] may present scalability issues for massive networks and the need for adjustments for different types of social networks, while the semi-supervised solution of [47] is dependent on large sets of labeled data. While both solutions can be used to detect user communities in urban spaces, as long as social interactions are adequately represented in the network, none of them explicitly addresses changes in community structures over time or issues related to data privacy, which could be an area of improvement for dynamic analysis.

Using different approaches, [48], [49] and [50] propose solutions to detect communities in complex networks. While [48] focuses on the difficulty of recognizing modular organization at a global level in large-scale networks, [49] aims to overcome the limitations of existing methods by considering both extrinsic and intrinsic signals and exploring the diverse relationships between nodes in a network, and [50] focuses on improving the accuracy and efficiency of the detection process. [48] develops a CNN-based method for classifying edges that is highly accurate in identifying community structures in real and synthetic networks; however, it does not address directed networks or group overlap. In turn, [49] proposes the HDMI framework for learning the representation of multiplex networks in a self-supervised way, using attention mechanisms. Results indicate that HDMI achieves state-of-the-art performance in supervised and unsupervised tasks, producing more discriminative

embeddings; however, it needs annotated data for supervised tasks. Also based on an attention method, [50] proposes a bipartite attention graph neural network model for detecting communities in supervised and unsupervised networks. The solution achieves improvements in classical and synthetic networks with a flexible approach that balances quality and speed, although there may be scalability restrictions for massive and complex networks. The three proposals can be adapted to detect user communities in urban spaces, as long as the corresponding network data is available. However, they do not discuss changes in communities over time or measures to protect data privacy.

With the aim of improving services and applications in smart cities through community detection, [55] compares the performance of different clustering algorithms on human mobility data to identify user communities in urban spaces. The work focuses on common mobilities features over spatio-temporal scales. Improvements are achieved in terms of both intra-community similarity and contact time. The work has limitations in terms of applicability in real environments and capturing all aspects of user communities. Continuing this study, [22] uses Deep Learning concepts to compare different classes of autoencoders in the process of extracting non-linear features from raw mobility data. Then, based on the application of a clustering algorithm on the data resulting from the previous process, the work can identify communities based on geographic preferences. Significant increases are achieved in both similarity and contact time between intra-community users, as well as an improvement in the identification of community structures. Unlike its precursor, the work considers the variation of communities over time. Both works, however, do not address issues relating to data privacy.

Based on network representation learning models, [53] and [54] present solutions to detect communities. Analyzing the activity profiles of adolescents in urban environments, [54] identifies high-quality communities consistent with racial and socioeconomic differences in urban neighborhoods. However, there is no ground truth for assessing the quality of these communities. In turn, [53] proposes Community-enhanced Network Representation Learning (CNRL) as a unified approach for community detection, which improves vertex classification, link prediction, and community detection in complex networks. Its limitations may include the need for adjustments for different types of networks and the computational complexity involved. Both approaches can be applied to detect user communities in urban spaces, as long as interactions between users are represented in a network. Regarding the evolution of communities over time, none of them explicitly address this issue, but the CNRL framework can be adapted to consider temporal dynamics. Data privacy policies are also not adopted.

Due to the scarcity of effective methods that capture changes in community

structures over time, [51] aims to improve community detection in dynamic networks by incorporating non-linear features using an evolutionary autoencoder. Although the method presents high accuracy and robustness, it presents high computational complexity, which can limit the scalability and practical applicability of the method in large-scale scenarios. The proposal can be applied to detect user communities in urban spaces, especially by considering the evolution of communities over time. However, the work does not directly address issues related to data privacy.

Finally, to perform clustering on urban data without compromising the privacy of participants' data, [52] proposes an deep embedded clustering method using Federated Learning to deal with privacy challenges in decentralized datasets. Both the feasibility of federated training compared to centralized approaches and its successful application in urban community detection are highlighted. However, the number of communities is predetermined in the model structure, which may restrict its adaptability at different time intervals, although this is not an issue addressed in the proposal. Furthermore, the work does not analyze the use of different federated aggregation algorithms or the impact of different amounts of mobility samples for model learning.

These works present significant results and contributions and can be used to detect communities in urban spaces. However, in addition to the limitations of each solution, often associated with scalability, none of them simultaneously addresses the temporal evolution of community structures and data privacy issues.

2.2 Autoencoders

Neural networks are a subset of Machine Learning and are the starting point for understanding Deep Learning. They are structures composed of layers of nodes – or units – connected to each other, where each unit has a weight, which is adjusted during model training. There are different types of neural networks for different purposes, such as prediction, categorization, dimensionality reduction, etc. A common subtype of these networks are so-called autoencoders.

An autoencoder is a neural network trained in an unsupervised fashion to summarize sample information into compact data structures and then reconstruct them into the original format [56]. This problem is formalized in [57] through the functions $A : \mathbb{R}^p \to \mathbb{R}^q$ (coding) and $Z : \mathbb{R}^q \to \mathbb{R}^p$ (decoding), which make up an autoencoder, to satisfy

$$\min_{A,Z} \sum_{i=1}^{N} \Delta(\bar{x_i}, x_i) \quad \text{where} \quad \bar{x_i} = Z(A(x_i)) \tag{2.1}$$

where Δ is the reconstruction loss function (O), which measures the distance

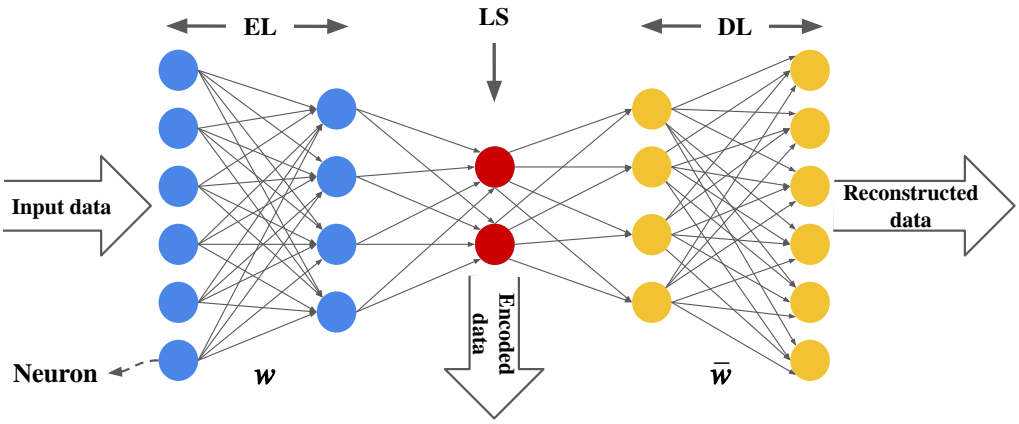


Figure 2.1: An autoencoder generates encoded representations from input data using EL weights w, and reconstructs it using DL weights $\bar{w}$.

between the input (x_i) and the generated output ($\bar{x}_i$) after decoding, for a total of N samples.

The structure of an autoencoder consists of encoding (EL) and decoding (DL) layers, with the latent space (LS) in between. EL is responsible for reducing the dimensionality of samples; LS consists of summarizing each input into an encoded representation; and DL is responsible for reconstructing samples from the LS, as illustrated in Figure 2.1. Each layer is composed of neurons – also called units or nodes – with weights (w in EL and $\bar{w}$ in DL) that are adjusted during model training. To generate the outputs of each layer, these neurons apply activation functions (AF), normally non-linear, to the weighted sum of the inputs. After passing each sample through the network during training, the corresponding output (reconstructed input) is compared to the original input so that the error is propagated to each node through the back-propagation algorithm [58]. Finally, a loss function is used to adjust the weights of the neurons in each layer.

Some recent works have achieved significant results by employing autoencoders in community detection tasks [22, 59–61]. In particular, [22] stands out for analyzing the performance of several autoencoder model architectures to learn non-linear mobility features from individual displacement data, capturing spatio-temporal geographic preferences useful for grouping users into communities. Traditional autoencoder (AE), variational autoencoder (VAE), convolutional autoencoder (CAE) and convolutional autoencoder with a fully connected layer (Full Convolutional Autoencoder, Full CAE or FCAE) were compared. It was found that FCAE is the most suitable autoencoder subclass to deal with the task of detecting user communities from mobility data in the form of heat maps, as it was the most successful in increasing the result of both the similarity and contact time metrics between users from the same community.

A Convolutional autoencoder (CAE), illustrated in Figure 2.2, is a specific subclass of autoencoders composed of convolution layers in the EL and deconvolution layers

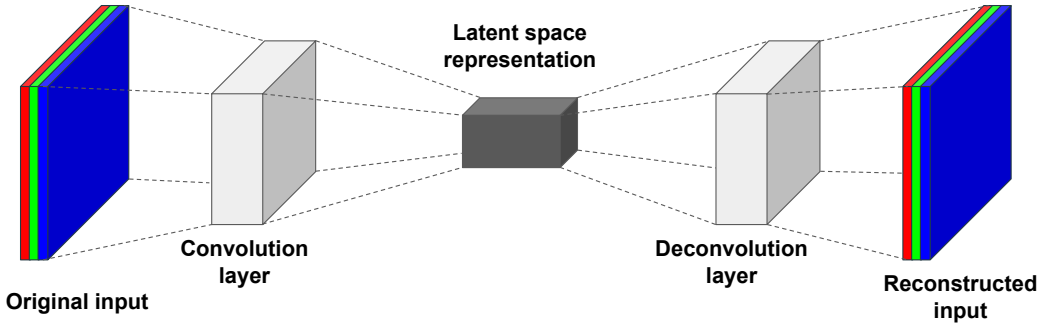


Figure 2.2: A convolutional autoencoder.

in the DL and is designed to process multidimensional vector inputs, such as images, for example [62]. When the latent space has a layer of fully connected nodes, it is called FCAE. Each convolution layer generates outputs called feature maps, capable of detecting and expressing local patterns that may be correlated in multidimensional vectors. For this, convolutional filters (or kernels) are used, whose quantity is equal to the total number of resulting feature maps [63]. The stride parameter controls the overlapping of operations while the filters slide over the vector, while the padding reduces the loss of information on the edges. These parameters and the filter size determine the output dimensions of each convolution layer. This size is especially important since the autoencoder aims to reduce the dimensionality of the original inputs. The operation of the convolution and deconvolution layers is described below.

2.2.1 Convolution Layers

Convolution is the operation performed between filters and the input of the convolution layer. This operation begins in the top left-hand corner of the input matrix, where convolution is performed on the submatrix of the same size as the filter used, as shown in Figure 2.3. The operation is the sum of the products of the elements located in the same positions of the two matrices. In this way, the convolutional process continues until the input matrix is completely traversed, as shown in Figure 2.4, resulting in the *feature map* shown in Figure 2.5.

The height $\bar{H}$ (or width $\bar{W}$) of the feature map can be calculated from the value of the corresponding dimension in the input matrix (height H or width W) and the kernel (height H_{kernel} or width W_{kernel}), the stride size s and padding p as shown in Equation 2.2.

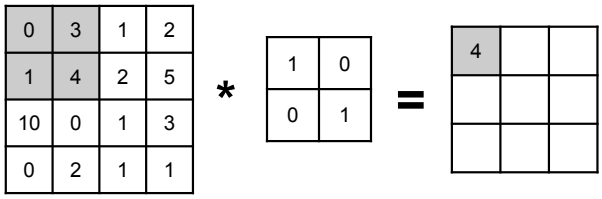


Figure 2.3: Start of the convolution process of a 4×4 pixel sample with a 2×2 filter and no padding.

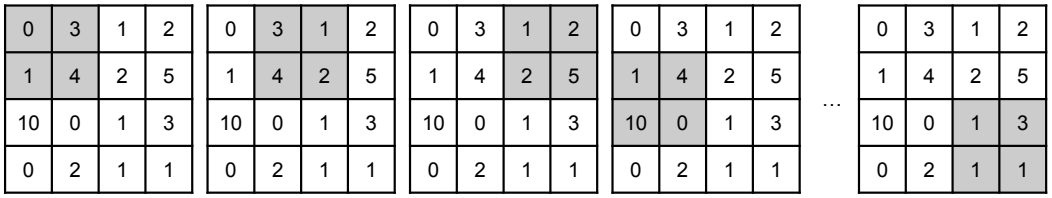


Figure 2.4: Convolution of a 4×4 pixel sample with a 2×2 filter, stride 1 and no padding.

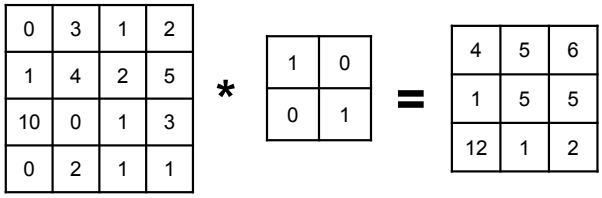


Figure 2.5: End of the convolution process of a 4×4 pixel sample with a 2×2 filter, stride 1 and no padding.

$$\bar{H} = \frac{H + 2p - H_{\text{kernel}}}{s} + 1$$
$$\bar{W} = \frac{W + 2p - W_{\text{kernel}}}{s} + 1 \tag{2.2}$$

Finally, the feature maps are processed by an activation function and reunited in the form of a three-dimensional matrix, whose dimensions are slightly different from the original input (smaller height and width and greater depth), resulting in the output of the convolution layer, as illustrated in Figure 2.6.

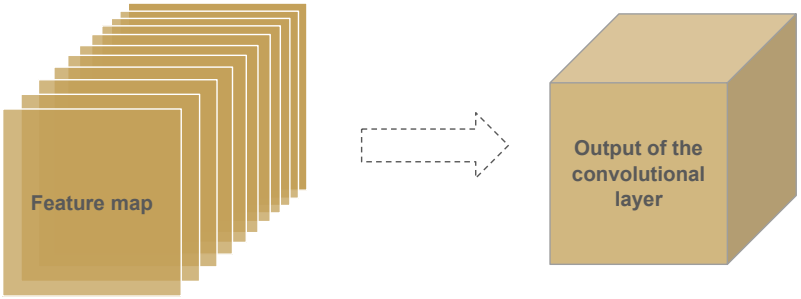


Figure 2.6: Output of the convolutional layer

Deconvolution layers

Deconvolution involves performing transformations in the opposite direction of the convolutional process. In other words, starting from something that has the format of the output of a convolution and producing something in the format of its input, a connectivity pattern compatible with such a process is maintained [64]. The layers used for this task, called transposed convolution layers, do not perform the deconvolution operation, as they do not produce outputs exactly the same as the inputs, but an approximation [65].

Each transposed convolution layer upsamples the feature map received as input using the same parameters mentioned above. Therefore, the output dimensions can be obtained from the formulations shown in Equation 2.3.

$$\begin{aligned} \bar{H} &= s(H-1) + H_{\text{kernel}} - 2p \\ \bar{W} &= s(W-1) + W_{\text{kernel}} - 2p \end{aligned} \tag{2.3}$$

Basically, the transposed convolution process comprises the operation of filters on each pixel of the feature map, so that a larger output is produced, as shown in Figure 2.7. The product of each pixel and the filter elements are assigned to the corresponding positions in the output matrix, as well as the sum of overlapping

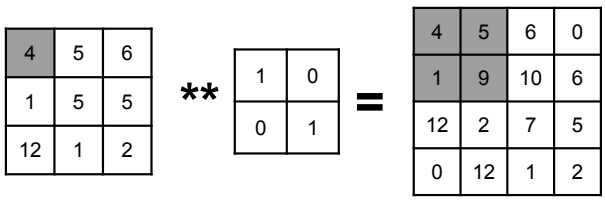


Figure 2.7: Upsampling a 3×3 feature map in a transposed convolution layer, with a 2×2 filter, stride 1 and no padding.

elements, whose occurrence depends on the stride value. Therefore, the training process is essential for obtaining filters that result in outputs that are as similar as possible to the inputs of the convolutional process.

Despite the valuable results obtained by solutions that use autoencoders to detect communities through the features extracted from the samples, it is important to consider that individual mobility data expresses daily routines and patterns of behavior. Therefore, they require some level of security against third-party access. In addition, as the volume of data increases, centralized training approaches can be affected by factors such as communication failures or time-consuming training, for example, which can hinder the operation of these solutions in real environments. These challenges have opened up space for distributed solutions, among which FL has stood out for its features such as privacy preservation and distributed collaborative learning [66–68].

2.3 Federated Learning

Due to growing concerns about privacy exposure, device availability, and possible network communication limitations, traditional model training methods based on data centralization have become impractical. In addition, the increased computing power of mobile devices has made it possible to use distributed learning techniques [69]. Among these techniques, Federated Learning (FL) has stood out as a paradigm that allows multiple clients to collaboratively train a shared model, as individual data is maintained locally [70, 71].

The optimization of federated models concentrates on some properties that differentiate FL from other conventional learning problems, such as non-IID distributions, the unbalanced number of samples in the data sets, the massive number of participants, and the possibility of low reliability communication links [70, 72].

During a conventional model training, one tries to minimize the $f(w)$ loss function using the w model and the sample set X, as shown in Equation 2.4. In FL, the client set U uses their own sample set for collaborative training, so this minimization is generalized according to Equation 2.5, where x^u is the sample set of user $u \in U$ [70].

$$\min_{w \in \mathbb{R}^d} f(w); \text{ where } f(w) \stackrel{\text{def}}{=} \frac{1}{|X|} \sum_{i=1}^{|X|} f_i(w) \tag{2.4}$$

$$f(w) = \sum_{u=1}^{|U|} \frac{|x^u|}{|X|} F_u(w); \text{ where } F_u(w) = \frac{1}{|x^u|} \sum_{i=1}^{|x^u|} f_i(w) \tag{2.5}$$

2.3.1 Data distribution and FL architecture

As each client uses a local dataset D, whose elements are defined by features F, labels L and identifiers I, FL is categorized based on the sample distribution characteristics of these sets [73, 74].

It is called horizontal FL (HFL) – or FL partitioned by sample –, the distribution scheme in which samples from different datasets overlap in their features but not in their identifiers, as shown in Figure 2.8(a). This architecture assumes the existence of honest participants and security against an honest-but-curious server, which is the only one capable of compromising customer privacy and data security [75, 76]. Formally, HFL can be expressed as follows:

$$F_i = F_j, L_i = L_j, I_i \neq I_j, \quad \forall D_i, D_j, \quad \text{where} \quad i \neq j$$

In turn, vertical FL (VFL) – or FL partitioned by feature – portrays the opposite situation. That is, there is overlap in the identifier space, but not in the feature space, as illustrated in Figure 2.8(b). VFL aggregates information from different features to collaboratively build a model from different sources. This architecture assumes the existence of two honest-but-curious participants and a third, semi-honest and independent, responsible for collecting intermediate (encrypted) results, gradient and loss calculation, and distribution of results to the two main participants [74]. VFL can be expressed as follows:

$$F_i \neq F_j, L_i \neq L_j, I_i = I_j, \quad \forall D_i, D_j, \quad \text{where} \quad i \neq j$$

Finally, in Federated Transfer Learning (FTL), there are not enough features or identifiers shared between samples from different datasets, so knowledge is transferred between domains, from the one with more resources to the one with less, as shown in Figure 2.8(c). In this architecture, both parties are considered honest-but-curious [77]. FTL can be expressed as follows:

$$F_i \neq F_j, L_i \neq L_j, I_i \neq I_j, \quad \forall D_i, D_j, \quad \text{onde} \quad i \neq j$$

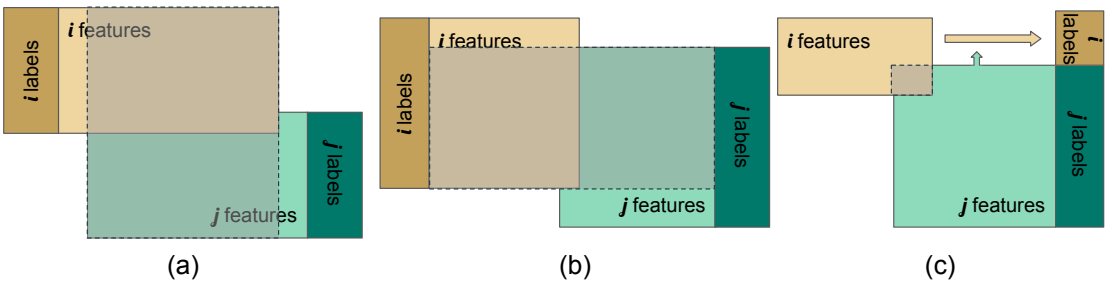


Figure 2.8: Data distribution in (a) HFL, (b) VFL and (c) FTL.

Through these categorizations, different FL architectures can be built, from systems based on a client-server scheme to systems based on a third-party collaborator, for example, [74]. In the first, multiple clients perform periodic local training orchestrated by a server that aggregates their local models into a global model. In the second, the collaborator assists two parties by sharing public keys and encrypted alignment to exchange gradients.

2.3.2 Privacy and security

One of the fundamental aspects associated with FL concerns data privacy, which is guaranteed through the construction of secure models. To achieve this, some techniques are typically adopted, such as Secure Aggregation and Differential Privacy.

Introduced in [75], the Secure Aggregation mechanism consists of a Secure Multi-party Computation (SMC) protocol that uses encryption to prevent the server from individually accessing updates from each device. This way, only the sum is revealed after receiving a sufficient number of updates. This ensures protection against honest-but-curious entities that may have access to instances in the aggregator. The protocol can be optionally enabled and is designed to handle a significant fraction of devices that may disconnect before aggregation is complete.

The Differential Privacy technique [78] adds noise to the data in order to distort fundamental attributes for identifying individuals. This makes its restoration unfeasible and guarantees the privacy of customers. This strategy, however, imposes a trade-off between model accuracy and individual data privacy [79], such that the more noise is added, the less accurate the model will become, especially when small data sets are used.

2.3.3 System heterogeneity

In the federated learning environment, it is common for there to be variability between the system characteristics of different devices, whether due to their hardware composition, network connectivity or energy capacity [80]. This variability, however, highlights the difficulty of some devices in keeping up with a certain level of interaction with the system.

In this context, synchronization policies ensure that the global model is updated equivalently and serially, so that it can be obtained periodically and equally by all client devices [70, 81]. However, adopting such policies results in solutions that are more susceptible to the impact of individual delays.

Alternatively, asynchronous policies try to reduce this impact in heterogeneous environments. However, since updates do not occur in an equivalent and serialized manner, the aggregate model is not accessed equally by all clients. Thus, a different

level of obsolescence can be observed in each device [82], resulting in a lower potential for generalization of the models [83]. Figure 2.9 illustrates the behavior of the two policies, the advantages and disadvantages of which must be evaluated when designing a solution.

2.3.4 Federated Aggregation

A typical model aggregation algorithm is called Federated Averaging (FedAvg) [70]. With it, the amount of computing is determined by three parameters: the fraction $\bar{U}$ of clients that contribute collaboratively in each training round r, so that a subset U_r of participants is created, where $|U_r| = \bar{U} \cdot |U|$ and U is the set of all users; the number of epochs E that these customers perform in each round; and the local minibatch size B used for each local update on the model.

Each participant $u \in U_r$ performs local updates on their local model w_r^u using a local loss gradient $g_u = \bigtriangledown \ell_u(w_r^u)$ and a learning rate η, according to Equation 2.6. These updates are reported so that the server calculates the weighted average of the resulting models, generating the new aggregated global model w_{r+1}, as established in Equation 2.7.

$$\forall u \in U_r, w_r^u \leftarrow w_r^u - \eta g_u \tag{2.6}$$

$$w_{r+1} \leftarrow \sum_{u=1}^{|U_r|} \frac{|x^u|}{|X|} w_{r+1}^u \tag{2.7}$$

Other popular aggregation algorithms are FedSGD [70] and FedProx [84]. While FedSGD is the baseline of federated learning, where in each round a single client is

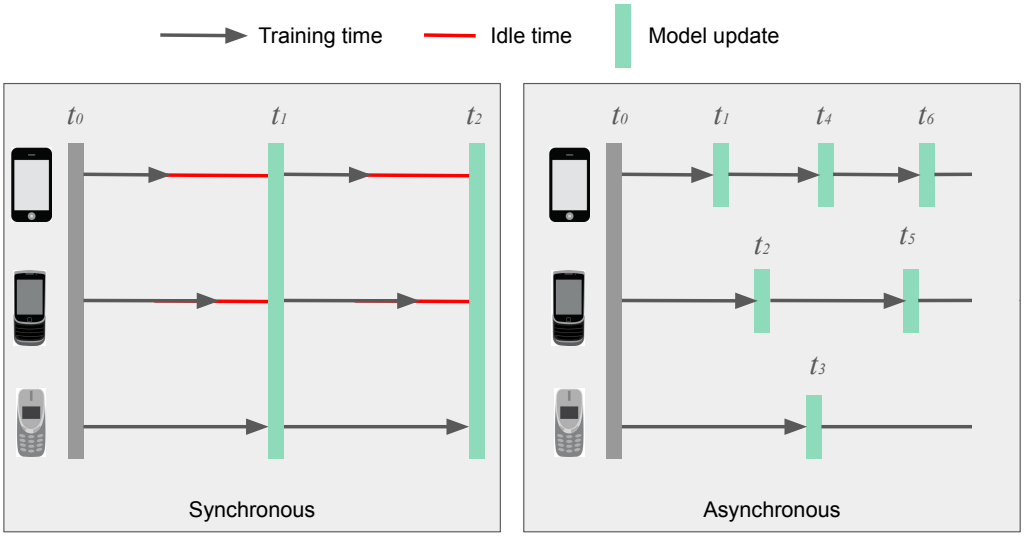


Figure 2.9: FL Training Policies

randomly selected to participate in the training process ($|U_r| = 1$), FedProx adds a proximal term in the objective function to deal with heterogeneity in FL. The purpose of this parameter is to regulate how far the local models deviate from the global model in each round. This term acts as a regularization that penalizes large deviations and promotes more stable convergence during local training. Unlike FedAvg, FedProx allows different devices to perform varying amounts of work locally (number of training epochs). This flexibility over participant effort is important since devices may have different computing capabilities, battery levels, and network connections. FedProx also aggregates partial solutions sent by devices and ensures convergence even in scenarios where data is generated in a non-identically distributed manner. The key parameters of FedProx are the parameter μ, associated with the proximal term in the loss function, and γ, which refers to the number of local epochs that each device performs during training. If $\mu > 0$, the proximal term penalizes large deviations from previous solutions, helping to force methods that might otherwise diverge to converge more stably. If $\mu = 0$, FedProx reduces to FedAvg, which does not have this regularization term. Equation 2.8 formulates the local updates to the model performed by FedProx.

$$\forall u \in U_r, w_r^u \leftarrow w_r^u - \eta g_u + \mu(w - w_r) \tag{2.8}$$

Algorithm 20 presents the pseudo-code of FedAvg and FedProx aggregation.

FL has been applied to solve various problems related to community detection, such as anomaly detection [85], predicting road flows based on the division of the local road network [68], and grouping customers based on purchasing behavior [86], among others. However, FL applied to the detection of user communities in urban space based on their mobility patterns and geographical preferences is still a less explored topic, especially when it comes to the periodic updates of such communities. So, in the next section, we propose our solution for periodically detecting user communities in urban spaces. To alleviate time-consuming training and communication costs, as well as to preserve individuals' mobility data privacy, we use FL to distribute feature extraction model training among multiple participants. Thus, the global model can capture different behavioral patterns to group such individuals after periodic and collaborative learning where there is no sharing of private data.

2.4 Clustering

According to [55], the Model-Based Clustering (MBC) algorithm [87] outperforms other algorithms such as *Hierarchical* [88], *K-means* [89] and *Fuzzy C-means* [90] in identifying communities from user mobility data.

Algorithm 1: FedAvg and FedProx Aggregation

Input: Users U; Fraction $\bar{U}$; Number of rounds R; Number of epochs (E or γ); Batch size B; Learning rate η

Server executes:
  initialize w_r
  for *each round* $r = 1, 2, 3 \dots R$ **do**
    $U_r \leftarrow$ (set of randomly selected $\bar{U} \cdot |U|$ users)
    for *each user* $u \in U_r$ **do**
      $w_r^u \leftarrow w_r$
      $w_{r+1}^u \leftarrow$ «*Runs FedAvgLocalTraining*(w_r^u) *or FedProxLocalTraining*(w_r^u)»
    $w_{r+1} \leftarrow$ Equation 2.7

FedAvgLocalTraining(w): *// Run on client u*
  $\mathcal{B} \leftarrow$ (splits x^u into batches of size B)
  for *each local epoch* $e = 1, 2, 3 \dots E$ **do**
    for *batch* $b \in \mathcal{B}$ **do**
      $w \leftarrow$ Equation 2.6
  return w

FedProxLocalTraining(w): *// Run on client u*
  $\mathcal{B} \leftarrow$ (splits x^u into batches of size B)
  for *each local epoch* $e = 1, 2, 3 \dots \gamma$ **do**
    for *batch* $b \in \mathcal{B}$ **do**
      $w \leftarrow$ Equation 2.8
  return w

Using a probabilistic approach, MBC assumes that data is generated from the mixture of a finite number of distributions with unknown parameters, where the probability of each element belonging to a cluster is computed. Thus, the mixture of Gaussian distributions – Gaussian Mixture Model (GMM) [91] –, used in [55], consists of the weighted sum of the Gaussian densities of K components, as established in Equation 2.9.

$$p(x|\lambda) = \sum_{k=1}^{K} w_k g(x|\mu_k, \Sigma_k), \tag{2.9}$$

where x corresponds to a D-dimensional vector that represents a sample, w_k the weights and $g(x|\mu_k, \Sigma_k)$ the densities of the Gaussian components. Each of these densities is equivalent to a D-varied Gaussian function, with mean μ_k and covariance matrix Σ_k, in the form

$$g(x|\mu_k, \Sigma_k) = \left((2\pi)^{D/2}|\Sigma_k|^{1/2}\right)^{-1} \exp\left\{-\tfrac{1}{2}(x-\mu_k)'\Sigma_k^{-1}(x-\mu_k)\right\}.$$

The mixture weights necessarily satisfy the constraint $\sum_{k=1}^{K} w_k = 1$, and the set of parameters passed to the model is represented by $\lambda = \{w_k, \mu_k, \Sigma_k\}$, with $k = 1, 2, ...K$.

GMM model selection

Different candidates from the GMM model can be obtained by varying the parameter K. The probabilistic selection methods Akaike Information Criterion (AIC) [92] and Bayesian Information Criterion (BIC) [93] assign a score to each candidate based on its performance and complexity, so that the "best" candidate is the one that obtains the lowest score. The AIC method is defined by Equation 2.10.

$$\text{AIC} = -2 \log L(\hat{\theta}) + 2d \tag{2.10}$$

where θ is the set of model parameters, $L(\hat{\theta})$ the likelihood of the candidate model on the data, and d the number of free parameters of the model. Through the formulation, the AIC assigns a lower score to the candidate who obtains the highest value in the likelihood function. However, the number of parameters is used as a penalty factor for the method.

In turn, the BIC method is defined in Equation 2.11.

$$BIC = -2 \log L(\hat{\theta}) + d \log (N) \tag{2.11}$$

where N is the number of observed samples. BIC applies a greater penalty to candidates with more parameters, while AIC favors performance, resulting in a less severe penalty for more complex models [94].

It is worth noting that, as the number of elements in the data set increases, the probability of the BIC method selecting the correct model also increases. On the other hand, the smaller the number of elements, the greater the probability that the method will choose simpler models [95].

Chapter 3

Federated Learning assisted framework to periodically identify communities

In this chapter, we present our framework for grouping individuals based on behavioral similarity detected from mobility data in urban spaces. Our work aims both to preserve individual privacy and to periodically update the state of communities. We use FL to ensure that samples from each individual are maintained locally and, at the same time, collaborate to improve the spatio-temporal mobility feature extraction model.

3.1 The problem of periodically identifying user communities in urban spaces

To present our proposal for periodically identifying communities, we first define a nomenclature:

1. **Displacement or mobility sample** (x): heat map that represents the user's mobility in a time interval t.
2. **User, individual or customer** (u): entity that generates samples and can be selected to participate in the training process. It is also the community's basic unit.
3. **Community**: group of individuals who share behavioral similarities.
4. **Encoded representation** ($\bar{x}$): code that represents the latent spatio-temporal features in the mobility sample.

The joint analysis of different users' mobility samples – referring to the same spatio-temporal scope – may indicate the existence of behavioral similarities among them due to their permanence or displacement in certain spaces within the same time interval. Thus, to highlight spatio-temporal features present in these samples, both to facilitate the detection of such similarities and to reduce the dimensionality of these data, a feature extraction technique [96] can be applied to them. As a result, a new, more concise collection of data is generated. Finally, when a clustering technique is applied to this collection, which represents the users' mobility, a set of subgroups is obtained, which correspond to communities based on behavioral patterns.

However, as these similarities are detected in individuals' behavior, which can vary over time, the composition and number of communities are subject to changes [12], making it essential to adjust these attributes as time progresses. For this, we assume that a new set of mobility samples is produced periodically, at each time interval t. Thus, for the feature extraction technique to be able to capture individuals' mobility characteristics from these samples, its parameters need to be adjusted at the same periodicity.

In this context, we propose using FL as the learning approach for the periodic adjustment of these parameters. Our choice is based on FL properties such as privacy preservation, collaborative learning and the possibility of processing samples as soon as they are released [26]. These characteristics can contribute to the evolution of the resource extraction model, preserving data privacy and facilitating the periodic identification of communities.

3.2 Raw Data Pre-processing

We initially assume that, using the Global Positioning System (GPS), each user can locally collect displacement information such as latitude, longitude and timestamp. GPS calculates the device's position at regular intervals, enabling recording this information with a high degree of precision and temporal frequency [38], making it an excellent option to capture details about individuals' displacement with low granularity. From these records, each individual can build a heat map representing their mobility within a time interval. Based on the methodology proposed in [22], the process can be established through the following steps:

1. **Spatial resolution setting**: the region where the individual may move is divided into a grid of height h and width w with cells of the same size. Within each cell, the user's permanence time is counted. In addition, if two users are in the same cell at the same time, the contact time between them is also counted.

$$DM = \begin{bmatrix} c_{0,0} & \cdots & c_{0,w} \\ c_{1,0} & \cdots & c_{1,w} \\ \cdots & \cdots & \cdots \\ c_{h,0} & \cdots & c_{h,w} \end{bmatrix} \longrightarrow$$

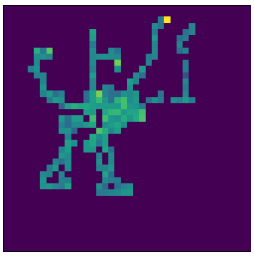


Figure 3.1: The Displacement Matrix (DM) is converted into a mobility heat map (sample), with each DM element corresponding to a pixel in the heat map.

2. **Temporal resolution setting**: it consists of establishing a size for the time interval t in which the GPS records are used to compose a new sample. This setting determines how often samples are generated, so decreasing the t size increases the sample generation frequency, resulting in the possibility of more frequent community state updating.

3. **Displacement Matrix (DM) filling**: the time spent by the user in each cell is added to a specific position c_{ij} in the DM, equivalent to the pixel identified by the indices i and j in the displacement heat map, as shown in Figure 3.1. Thus, the longer the user stays in the cell, the greater the intensity of the corresponding pixel. After filling in completely, each value is normalized by the user's maximum travel time within the interval t.

4. **Logit transformation**: To highlight similarities among corresponding variables in user samples, the *Logarithmic Likelihood* (Logit) [97] transformation is applied on DM. This non-linear transformation maps all values in the DM – which range from 0 to 1 – to values in the scale (-∞, ∞), symmetrical in 0.5. Assuming that p is the probability of an event happening and $(1-p)$ is the probability of the same event not happening, the relative probability of events occurring is given by $p/(1-p)$. Thus, as the Logit function is defined as a logarithm of relative probabilities, it is given by $\text{Logit}(p) = \log\left(\frac{p}{1-p}\right)$. As this function does not define a result when $p = 1$ or $p = 0$, which may occur if the user remains in the same cell during all the records made in t ($p = 1$), or when no record occurs during the same interval ($p = 0$), a small value ϵ is added to the parts of the fraction, as suggested in [55]. This value is set as $\epsilon = 10^{-6}$.

It is worth mentioning that, if for any reason a user fails to collect raw mobility data during an interval t, the heat map referring to that same interval will be empty. Such samples are disregarded in the feature extraction process.

3.3 Mobility Features Extraction Model

To extract spatio-temporal mobility features from individual samples, we use an autoencoder model based on the methodology proposed in [22]. According to this work, the Full Convolutional Autoencoder (FCAE) model is a more appropriate subclass of autoencoders for extracting spatio-temporal mobility features from data in the form of heat maps since, in comparison with other architectures, it was the one that managed to increase the metrics of similarity and contact time between users of the same community. Thus, we concentrated our efforts on elaborating and training a model corresponding to this subclass.

The architectural settings for building FCAE models may vary for different scenarios. This is due to the parameterization of the raw data pre-processing steps, especially regarding the spatial resolution setting, since the models depend on the size of the inputs. In other words, in scenarios of greater spatial extent, where more displacement patterns can be detected, autoencoder models with more layers and filters are more appropriate for capturing these latent features. Next, we describe how the EL, LS and DL layer sets were configured.

3.3.1 Encoding Layers (EL)

The first layer in the FCAE model is an input layer, which receives the samples and detects their format in terms of height (H), width (W) and depth (relative to the RGB channel) [98]. As the user mobility samples are grayscale images, in which each pixel provides only information about its intensity, they have a depth equal to 1. Thus, the input layer is parameterized with the values $(H, W, 1)$, as shown in Figure 3.2. Next, convolution layers (Conv2D) are added. These layers highlight latent features of the input samples and reduce their dimensionality using specific

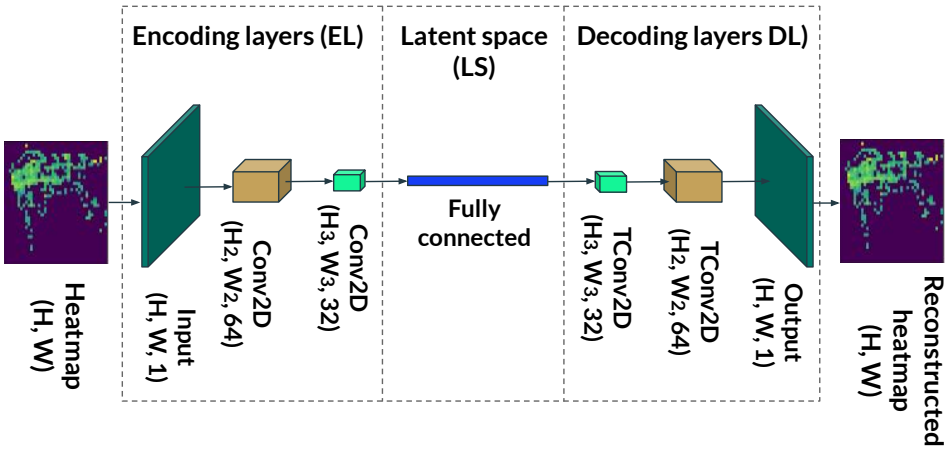


Figure 3.2: FCAE model architecture for extracting latent mobility features from user samples.

parameters, such as kernel size, stride and padding [63]. As output, each convolution layer produces 3D vectors with new dimensions. Through this process, behavioral information, such as geographical preferences and displacement patterns, can be detected. Finally, a flattening layer is connected to the last convolution layer to realign the output 3D vector elements into a 1D vector.

3.3.2 Latent Space (LS)

A fully connected (or dense) layer is added with a reduced number of nodes, where each one is connected to all the nodes of the previous layer. Each output produced by this layer is a 1D vector that corresponds to the encoded representation ($\bar{x}$) of a user mobility sample.

3.3.3 Decoding Layers (DL)

The first layer in this set is dense, whose number of nodes corresponds to the size of the 1D vector produced by the flattening layer. Next, a reshaping layer realigns the incoming elements into a new 3D vector with the same dimensions as the one produced by the last convolution layer. To reverse the convolution process, transposed convolution layers (TConv2D) are added to the sequence. Each one upsamples its input using the same parameters mentioned above, maintaining a standard connectivity compatible with the previous process [64].

After going through all these layers, the final FCAE model output corresponds to the heat map reconstructions. These outputs, in turn, are compared to the original samples, and the error is propagated to the nodes in each layer through the back-propagation algorithm [58] for parameter adjustment during model training. The training process, therefore, consists of minimizing the loss obtained between the original sample and its reconstruction. For this, the Mean Squared Error (MSE) is adopted as a loss function.

3.4 FL-based Approach

To learn the parameters of the FCAE model, we adopted FL as the training method. Thus, we want to take advantage of FL characteristics that can contribute to the evolution of the feature extraction model while preserving the privacy of the data. Next, the FL architecture, the local training process, the global model aggregation and the pseudocode of our framework are presented.

3.4.1 Data distribution and FL architecture

To define the training architecture, we first need to understand the data distribution. For this, we consider that all users are restricted to moving within the same spatial scope (a city, for example). With this, the data features are the same for all datasets (i.e., each cell where these users move is a feature). On the other hand, each user maintains data regarding its own displacement, i.e., there is no sharing of sample identifiers between entities. Thus, it can be said that the different datasets, although they share the same feature space, do not share the same sample space.

Therefore, we use the Horizontal FL (HFL) architecture, also referred to as FL partitioned by sample [99], in which datasets share overlapping feature space but differ in samples. In a typical client-server HFL architecture, participants train the model individually with their local data and are assisted by a central entity (a server) to aggregate the resulting parameters and produce a global model. In the experiments to be presented in the next chapters, we use the FedAvg [70] and FedProx [84] aggregation algorithms separately in our framework in order to compare their performance and influence on community detection.

3.4.2 Local training process

As mentioned before, the objective of the training process is to find the model parameters capable of minimizing the loss function (in this case, the MSE function). Since the FCAE model is composed of a part that encodes ($A : \mathbb{R}^p \rightarrow \mathbb{R}^q$) and another that decodes ($Z : \mathbb{R}^q \rightarrow \mathbb{R}^p$) the samples, the loss function during the local training performed by the user $u \in U_r$ is established according to Equation 3.1, where x^u is the sample set of user u, while a_w and $z_{\bar{w}}$ are the coding and decoding gradients associated with w and $\bar{w}$ weights, respectively, and x_i^u represents a mobility sample. In addition, when performing the local training in a round r, each user u performs M_r^u updates on the FCAE model weights using the parameters E, x^u and B, as established in Equation 3.2.

$$\ell^u = \frac{1}{|x^u|} \sum_{i=1}^{|x^u|} ||z_{\bar{w}}(a_w(x_i^u)) - x_i^u||^2 \tag{3.1}$$

$$M_r^u = E\frac{x^u}{B} \tag{3.2}$$

In turn, with each local update, the weights of the FCAE model (w^u) are modified according to the local training method. Lines 22 to 31 of Algorithm 2 describe the local training process. Aiming to investigate the impact of aged samples on the model's performance, we propose two time window-based strategies for processing samples during local training: the Accumulative Window and the Sliding Window.

Accumulative Window (ACC) This strategy consists of using all the non-empty samples in the user's data set up to the current time interval t at which the FCAE model is trained. That is, local training during a round r considers all non-empty heat maps produced up to t, so the set of training samples is defined by $x^u = \{x_i^u | 0 \leq i \leq t\}$, where x_i^u corresponds to the non-empty sample produced at interval i. Figure 3.3(a) illustrates the ACC window-strategy, in which all non-empty samples from any time interval are used for local training.

Through this strategy, in each round, any user with at least one mobility sample, produced at any time interval, can be selected to update the FCAE model, whose weights will be aggregated with those of other clients to compose the global model. However, since the total local updates is dependent on the number of samples available for training, different levels of user participation can be achieved, resulting in an eventual imbalance of user contributions.

Sliding Window (SLI) This strategy uses at most sw samples for local training. Thus, the possession of non-empty samples within a window of size sw is established as a criterion for the participation of users, so that if $sw = 3$, for example, then the set of training samples is defined by $x^u = \{x_i^u | t - 3 < i \leq t\}$ where x_i^u corresponds to the non-empty sample produced at interval i. Thus, the size of x^u is bounded to $|x^u| \leq sw$.

The SLI strategy limits the number of past samples that will be used for local training. In addition, it is mandatory that, to update the model, the user does not utilize more than sw samples. With this, the SLI window-strategy restricts the

(a) ACC window-strategy (b) SLI window-strategy

Figure 3.3: In the ACC window-strategy (a), all non-empty samples are used to train the model. When there are no such samples at recent intervals, the old ones are used. In the SLI window-strategy (b), no more than sw recent non-empty samples are used to train the model. When there are no such samples in the window of size sw, the user does not contribute updates to the model. In this example, non-empty samples are marked with a green border, whereas empty ones have a red border and $sw = 3$.

participation of users who, for some reason, fail to record mobility data within the most recent sw window or who only have samples considered obsolete, as shown in Figure 3.3(b). In this way, we seek to reduce the imbalance between the contributions by limiting the number of updates for each user (M_r^u).

To represent a training window (whether ACC or SLI), we use the notation of closed and open sets. For example, at interval t_2 the model is trained with window $[0, 3)$ and processes samples from intervals t_0, t_1 and t_2 (in this case, if the strategy is SLI, then $sw = 3$). The use of window strategies ACC and SLI is described in lines 23 to 26 of Algorithm 2.

3.4.3 The FCAE Global Model Aggregation

For each interval t, R rounds of local training are orchestrated by the server, so that each round r results in M_r updates, where $M_r = \sum_{u=1}^{|U_r|} M_r^u$. When starting a new training round r, the server communicates the current global model weights (w_r) to all participants, who in turn update their local models (w^u), so that $w^u \leftarrow w_r$. At the end of each round, the server takes care of aggregating, through a weighted average, the weights w_{r+1}^u from each participant u, to generate the global model for the next round $(r + 1)$, as defined by Equation 3.3. In this way, the global FCAE model evolves as time progresses, with the t granularity being the factor that determines the scale of progress. Lines 2 to 9 of Algorithm 2 describe the training orchestration process carried out by the server. Figure 3.4 illustrates the process from raw data collection to the local training and global model aggregation stages.

$$w_{r+1} = \frac{1}{\sum_{u=1}^{|U_r|} x^u} \sum_{u}^{U_r} x^u w_{r+1}^u \tag{3.3}$$

A factor of great importance in FL is the preservation of individual privacy. Thus, we need to prevent the encoded representations from being easily decoded by third parties who have access to the global model. Therefore, considering the possibility of leaking private information by inspecting shared models [99–101], we propose that only the EL and LS weights (w) be communicated to the server, while the DL ones ($\bar{w}$) are maintained locally.

Thus, when selected to participate in a round, the users receive updated EL and LS weights from the server, while the DL ones are initialized with random values – if it is an initial round – or with values from the last round. During local training, these users update w and $\bar{w}$, but they only communicate to the server the EL and LS weights, which are aggregated with those from other users to compose the global model.

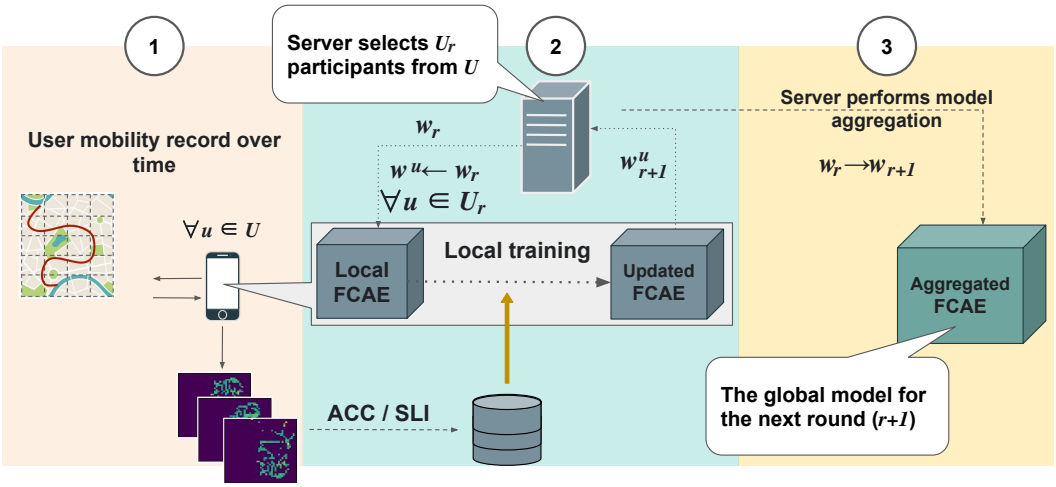


Figure 3.4: Firstly, each user periodically produces mobility samples from GPS records. Next, the federated training rounds start. When a new round r begins, the server transfers the current global model weights (w_r) to update the local model (w^u) of users in U_r, so that $w^u \leftarrow w_r \forall u \in U_r$. The users in U_r train the local FCAE model using a predetermined window-strategy (ACC or SLI), updating the local weights to w^u_{r+1}, which are sent to the server. Finally, the server performs the global model aggregation for the next round (r+1).

3.5 Communities Identification

To classify individuals moving around a city according to their behavioral similarity, a grouping technique must be used. Since the data used is not labeled, unsupervised learning approaches are the most appropriate to be employed over them [102], especially given the lack of prior knowledge about the relationship between individuals.

Thus, the Gaussian Mixture Model (GMM) [91] algorithm, which obtained the best results in relation to the metrics evaluated in [55] when compared to other clustering techniques, is applied to the encoded representations generated periodically by the FCAE model. This algorithm assumes that the inputs are generated from a mixture of a finite number of Gaussian distributions with unknown parameters, so that the probability that each element belongs to a cluster is computed. For the GMM, the number K of clusters (i.e., communities) is required.

3.5.1 Setting the Number of Communities

As previously established, the number of communities in which individuals are satisfactorily allocated may vary over time for a given scenario. Thus, we propose that, at each interval t, different values be assigned to the parameter K. Thus, each parameterization results in a distinct set of communities, whose element's distribution must be evaluated to define the best fit – that is, the best value for K – for that particular interval.

To perform such an evaluation, we use the analytical scoring methods Akaike Information Criterion (AIC) [92] and Bayesian Information Criterion (BIC) [103] and compare their results. In both methods, a score is assigned to each candidate under evaluation, so that the "best" model is the one that obtains the lowest score. The BIC method tends to penalize candidates with more parameters (higher complexity), while the AIC method favors candidates that produce greater value in the likelihood function (better performance), resulting in a less forceful penalty on more complex models [94]. The use of the AIC and BIC methods is described in lines 13 to 20 of Algorithm 2.

It is worth mentioning that, as the number of elements in the data set increases, the probability of BIC selecting the correct model also increases. In contrast, the smaller the number of elements, the greater the probability that this method will choose simpler models [95].

Periodic updating of community sets

To update the communities as time progresses, we propose that, for each interval t, each user generates an encoded representation of their mobility using the current global FCAE model, and then sends it to the server so that the clustering algorithm is applied to the representations of all users in that same interval. To reduce the risks of leaking each individual's private information, the process described in Section 3.4.3 is fundamental, so that the encoded representations of mobility that a user sends to the server cannot be decoded by third parties. Finally, after community identification, the server can make the best use of the information about the similarity relationship among individuals as time progresses. Lines 10 to 21 of Algorithm 2 describe the clustering process.

Algorithm 2: FL ASSISTED FRAMEWORK TO PERIODICALLY IDENTIFY USER COMMUNITIES

Input: Number of users U; Fraction of participating users $\bar{U}$; Number of rounds R; Number of epochs E; Batch size B; SLI window size sw; Max number of communities K; Learning rate η

Output: communitiesAIC, communitiesBIC

Server executes:
- initialize w_r
- **for** *each interval* $t = 0, 1, 2, \ldots$ **do**
  - **for** *each round* $r = 1, 2, 3 \ldots R$ **do**
    - $U_r \leftarrow$ (randomly selects $\bar{U} \cdot U$ users)
    - **for** *each user* $u \in U_r$ **do**
      - $w^u \leftarrow w_r$
      - $w^u_{r+1} \leftarrow$ **LocalTraining**(t, w^u)
    - $w_{r+1} \leftarrow$ Equation 3.3
  - initialize list L of encoded representations
  - **for** *each user* $u \in U$ **do**
    - Add **EncodedHM**(t, u) to L
  - **for** *each candidate* $k = 2, 3, \ldots K$ **do**
    - Run GMM(k, L)
    - Store community labels
    - Store AIC and BIC scores
  - choiceAIC $\leftarrow k$ linked to min(AIC scores)
  - choiceBIC $\leftarrow k$ linked to min(BIC scores)
  - communitiesAIC $\leftarrow$ labels from choiceAIC
  - communitiesBIC $\leftarrow$ labels from choiceBIC
  - **return** communitiesAIC, communitiesBIC

LocalTraining(t, w^u): // *Run on user* u
- **if** *window strategy is ACC* **then**
  - $X^u = \{x^u_i | 0 \leq i \leq t\}$
- **else**
  - $X^u = \{x^u_i | t - sw < i \leq t\}$
- $\mathcal{B} \leftarrow$ (splits X^u into batches of size B)
- **for** *each local epoch* $e = 1, 2, 3 \ldots E$ **do**
  - **for** *batch* $b \in B$ **do**
    - $w^u \leftarrow$ Equation 2.6 (FedAvg) or Equation 2.8 (FedProx)
- **return** w^u

EncodedHM(t, u): // *Run on user* u
- **return** the encoded heat map of user u at interval t

Chapter 4

Experimental Methodology

In this chapter, we present the experimentation methodology to evaluate the performance of our solution. Thus, the scenarios and model structures created, the evaluation metrics and the training parameters used are presented. Our experimental evaluation is motivated by the following research questions (RQ):

RQ1: What are the implications of using the probabilistic methods AIC and BIC to define the number of communities (K)?

RQ2: What are the impacts of ACC and SLI window-strategies for federated FCAE model training and communities' composition?

RQ3: Is there a significant difference between the results obtained by FL-based and centralized approaches for the FCAE model in identifying communities?

To answer these questions, we set up three experimental scenarios from three real mobility datasets so that we could evaluate our solution. For each of these scenarios, we pre-processed the data to generate the mobility heat map for each user in each time interval. We then calculated the performance metrics (see Section 4.2) for each pair of users in each interval to determine the baseline for comparison in relation to the communities subsequently detected. At each time interval, the model was trained using a window-based strategy (ACC or SLI). At the end of training, the model generated encoded representations of the heat maps of the current interval, which were used as input for the clustering algorithm. We used multiple values of K (number of communities) in the clustering process and delegated the choice of a value among the candidates to the AIC and BIC methods. Each encoded representation (and user) is then assigned a community label. With these labels and the value of the metrics for each pair of users, we calculated the average and confidence interval of the same metrics for intra- and inter-community users.

Next, these datasets are described, as well as the pre-processing performed on them. Also, the model structure, the hyperparameter settings, and the description

of the evaluation metrics are presented. Our entire system was implemented using TensorFlow [80].

4.1 Experimental Datasets

To represent the individuals' behavior in the urban space, the San Francisco Cabs [104], Roma Taxi [105] and Next Generation Simulation (NGSIM) Vehicle Trajectories and Supporting Data [106] datasets were used, which are briefly described below.

1. **San Francisco Cabs (SFC)** is a vehicular mobility dataset consisting of trajectories taken over 24 days (between May 17th and June 10th, 2008) by 536 taxis in the city of San Francisco, USA. GPS logs are taken at intervals ranging from 1 to 3 minutes. Vehicle identification, latitude, longitude, date/time stamp, and vehicle occupancy status are the attributes of each record, the latter being disregarded in our experiments.

2. **Roma Taxi (RT)** is a vehicular mobility dataset consisting of trajectories taken over 30 days (between February 1st and March 2nd, 2014) by approximately 320 taxis in the city of Rome, Italy. GPS logs are taken at approximately 7 second intervals. Vehicle identification, latitude, longitude and timestamp are attributes of each record.

3. **Next Generation Simulation (NGSIM)** is a vehicular mobility dataset consisting of trajectories taken on southbound US 101 and Lankershim Boulevard in Los Angeles, CA; eastbound I-80 in Emeryville, CA; and Peachtree Street in Atlanta, Georgia. Each subset of trajectories has characteristics of both the collection and the roadways themselves, as shown in Table 4.1. We verified that the US 101 subset is the only one in which the large number of vehicles is concentrated in a short temporal scope, in which mobility occurs throughout all time intervals. Therefore, we chose to work with this subset, referring to it as the NGSIM scenario.

Table 4.1: Subsets of the NGSIM dataset identified by highway name

Subset	Duration	Width	Length	Total vehicles
US 101	46 minutes	76 m	2.24 km	2847
Lankershim	32 minutes	221 m	1.66 km	1506
I-80	33 minutes	97 m	1.80 km	3001
Peachtree	12 days and 2 hours	487 m	2.12 km	1545

4.1.1 Spatial resolution setting

For the SFC and RT datasets, we delimited the mobility scope as areas of approximately $12km$x$12km$. The displacement matrix was constructed and completed by subdividing such areas into cells of $300m$x$300m$, resulting in samples (heat maps) of 40 pixels x 40 pixels. It is worth noting that the cell granularity choice has a direct impact on the heat map resolution. Therefore, it is important to balance this parameter so that it is not so low as to lose important mobility information or so high that there is no coexistence of individuals in the same cell at a given time interval. Table 4.2 presents the coordinates for delimiting the spatial scope in the SFC and RT scenarios. In turn, the $76m$x$2.24km$ area of the NGSIM scenario was subdivided into cells of $40m$x$40m$, resulting in samples of 2 pixels x 56 pixels. However, to calculate the similarity metrics, we added an extra row of null pixels above and below each sample, thus obtaining heat maps of 4 pixels x 56 pixels. It is worth noting that spatial resolution directly influences the average contact time between individuals, since to obtain this value, the coexistence of users in the same cells must be verified.

Table 4.2: Latitude and longitude (minimum and maximum) used to delimit the mobility scope for the SFC and RT scenarios

Dataset	Min. Latitude	Max. Latitude	Min. Longitude	Max. Longitude
SFC	37.71000	37.81399	-122.51584	-122.38263
RT	41.84250	41.94607	12.42272	12.56157

4.1.2 Temporal resolution setting

To define the spatial resolution, we experimented with different sizes for the interval t, in order to reduce the number of users who failed to collect mobility records in each interval. Such failures, as previously explained, generate empty heat maps, which are disregarded because they do not present information about mobility or permanence in cells. Reducing the size of t (high temporal resolution) would imply more frequent community updates, so that the behavioral similarity among users would be portrayed with low obsolescence.

We observed in the SFC and RT datasets that the sharp decrease in the size of t implies a drastic reduction in the number of customers whose location is recorded in each interval. This suggests that, in both cases, there is a low tendency for individuals to record their location at the same or close instants of time. On the other hand, in the NGSIM scenario, we observed the possibility of reducing t to the scale of a few minutes while maintaining a significant number of non-empty samples in each

interval. Table 4.3 presents the size of t assigned to each scenario. For the SFC scenario, we extracted results from a temporal scope equivalent to one week, from May 27 to June 3, 2008. For the RT scenario, the temporal scope was limited to one workweek, from February 10 to 15, 2014. For the NGSIM scenario, the temporal scope includes all mobility records available in the dataset, that is, those made during 46 minutes on June 15, 2005.

Table 4.3: Size of the interval t and number of intervals per scenario

Dataset	Size of the interval t	Number of intervals
SFC	4 hours	42
RT	4 hours	30
NGSIM	5 minutes	10

4.1.3 Feature extraction model structure

Since the heat maps of the SFC and RT scenarios have the same dimensions, being differentiated – in terms of pre-processing – only by temporal resolution, we used the same FCAE model structure in both scenarios. Thus, the input layer of the model is parameterized as $W = 40$, $H = 40$, and $D = 1$. Next, the model sequences three convolution layers with 128, 64 and 32 kernels of size $3x3$, which use the *Rectified Linear Unit* (ReLU) as the activation function, where $ReLU(x) = max(0, x)$. The fully connected layer uses the same activation function (AF) and is composed of 100 units, resulting in a vector with the same number of elements, which is the encoded representation size. Finally, the transposed convolution layers, which are symmetrical to the convolution layers, have 32, 64 and 128 kernels, respectively, and also use ReLU as activation function.

Table 4.4: The FCAE model structure and attributes for the SFC, RT and NGSIM scenarios

Parameter	SFC and RT	NGSIM
Input shape	40x40x1	4x56x1
Depth	3	2
Encoder kernels	128-64-32	16-8
Latent space	100	20
Decoder kernels	32-64-128	8-16
Kernels' size	3x3	5x5
Stride	2	2
Padding	1	1
Activation Function	ReLU	ReLU
Loss Function	MSE	MSE

For the NGSIM scenario, the structure of the FCAE model consists of an input layer with $W = 4$, $H = 56$, and $D = 1$. It has two convolution layers, the first with

16 and the second with 8 kernels, all $5x5$ in size, and two transposed convolution layers, symmetrical to the convolutional ones. The ReLU is used as AF in both the encoding and decoding layers and, finally, the latent space is made up of 20 units. Table 4.4 summarizes the FCAE model structure in relation to the sequence of layers and their attributes.

4.1.4 Hyperparameter Settings

To configure the hyperparameters, we experimented with different combinations of values using the FedAvg and FedProx aggregation algorithms before making the final assignment. To evaluate the performance of each combination, we analyzed the result of the loss function on the global model at the end of each round of federated training in each scenario. Thus, Figure 4.1, 4.2 and 4.3 illustrate the aforementioned losses when the number of epochs is $E = 3$, $E = 6$, $E = 9$, and the number of rounds is $R = 30$. When analyzing the number of rounds, we noticed that the first training window – i.e., [0, 1), when each user only has one sample – presented the most considerable variation in losses between two consecutive rounds. Also, that after a certain number of rounds, the losses did not vary much from one round to

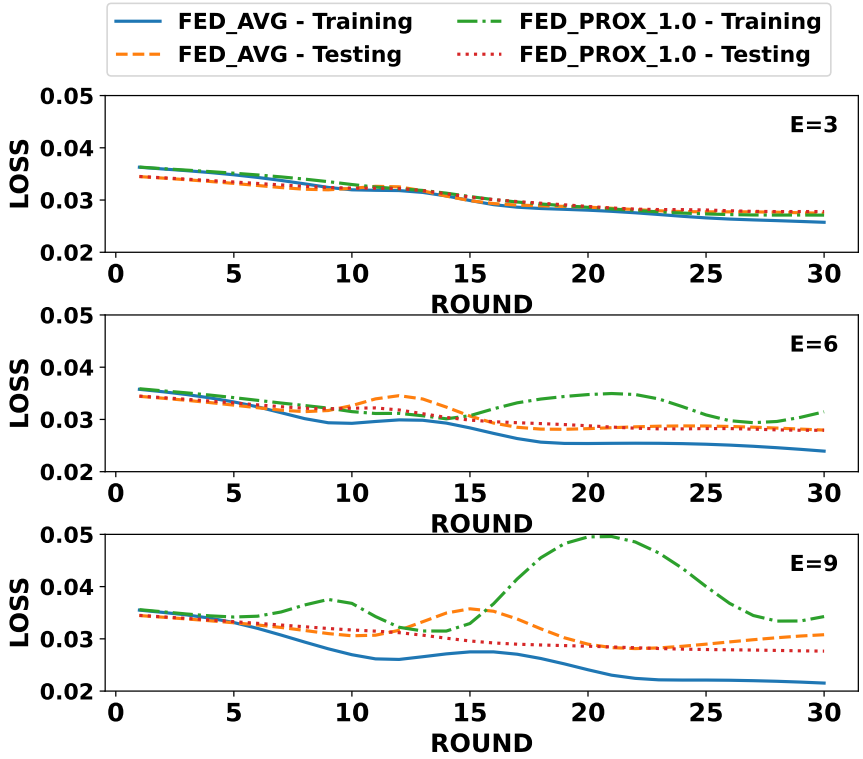


Figure 4.1: The FedAvg and FedProx training and testing losses obtained in the SFC scenario with data from the window $[0, 1)$ (for training) and the window $[1, 2)$ (for testing), when the number of epochs is assigned the values $E = 3$, $E = 6$, and $E = 9$, and the number of rounds is fixed at $R = 30$.

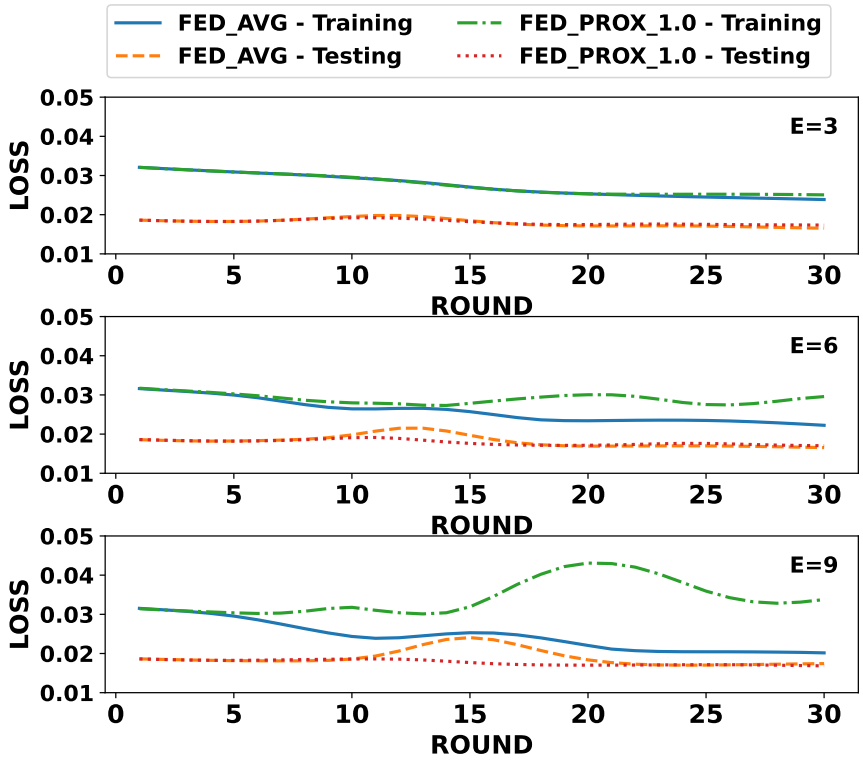


Figure 4.2: The FedAvg and FedProx training and testing losses obtained in the RT scenario with data from the window $[0, 1)$ (for training) and the window $[1, 2)$ (for testing), when the number of epochs is assigned the values $E = 3$, $E = 6$, and $E = 9$, and the number of rounds is fixed at $R = 30$.

the next. In subsequent training windows, in which the number of samples increases, we noticed an even lower variation from the start to the end of training. Thus, we set the number of rounds for all intervals as $R = 20$.

Similarly, we analyzed the local training parameters, such as epochs. The experiments performed in the three scenarios show that training with fewer epochs not only kept the loss quite similar between the two algorithms throughout the rounds but also resulted in a loss quite close to FedAvg, which obtained the lowest loss when E = 9. Therefore, we decided to set $E = 3$.

Regarding the learning rate, we experimented with values equal to 0.003, 0.001 and 0.0005 for the SFC and RT scenarios. Unlike the first two settings, only the last one resulted in a slope on the resulting loss. In turn, we observed that these values were not suitable for training the NGSIM scenario, either because they caused too many variations or a low slope in the loss curve. We therefore applied a learning rate of 0.002 for this scenario. We also defined the SLI window-strategy size as $sw = 3$. Finally, for the choice of K by the AIC and BIC methods, we defined a set of candidates that varies from 1 to 20 for the SFC and RT scenarios and from 1 to 30 for the NGSIM scenario.

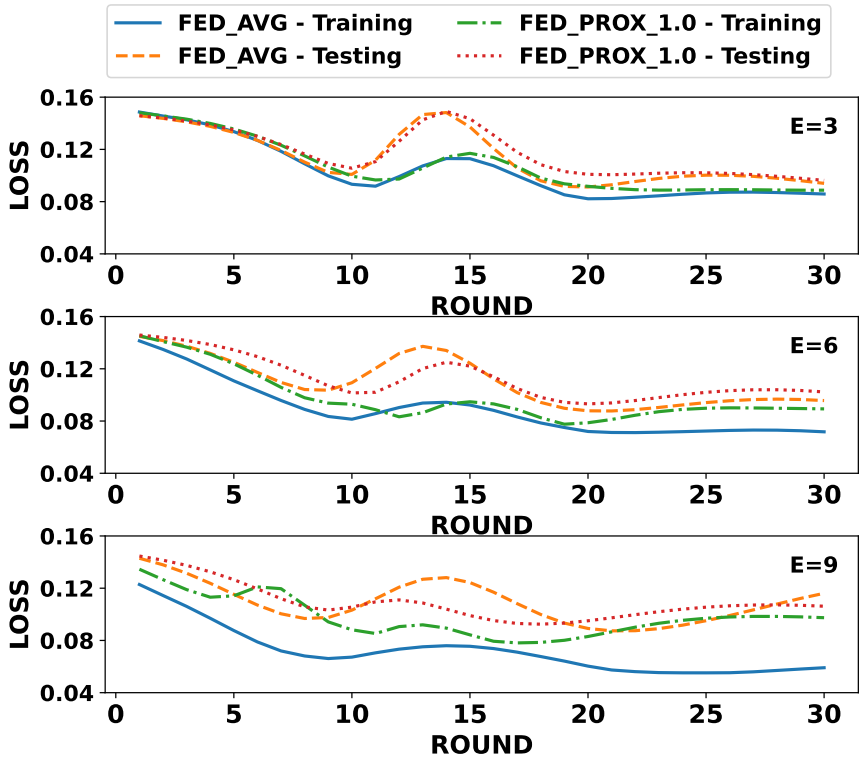


Figure 4.3: The FedAvg and FedProx training and testing losses obtained in the NGSIM scenario with data from the window $[0, 1)$ (for training) and the window $[1, 2)$ (for testing), when the number of epochs is assigned the values $E = 3$, $E = 6$, and $E = 9$, and the number of rounds is fixed at $R = 30$.

To conduct experiments in a centralized way, in which the server is solely responsible for training the model, we set training parameters for an FCAE model with the same structure defined in Section 4.1.3. To perform a fair comparison, the parameters were set for the centralized experiment so that the number of model updates was equal for the corresponding intervals on the FL-based approach. Equation 4.1 expresses how the number of updates in each interval is computed, where E is the number of epochs, N the number of samples – which varies at each interval but is the same for both approaches –, B the batch size, and R the number of rounds (if the approach is FL-based, otherwise $R = 1$). The Table 4.5 presents the hyperparameter settings for the FL-based and centralized approaches.

$$M = \left(E \frac{N}{B} \right) R \tag{4.1}$$

4.2 Evaluation Metrics

To measure the communities' quality, we used some metrics, as suggested in [22], to verify the similarity and dissimilarity degrees among users from the same community

Table 4.5: Hyperparameter settings for the FL-based and centralized approaches for SFC, RT and NGSIM scenarios

Parameter	SFC and RT		NGSIM	
	FL-based	Centralized	FL-based	Centralized
Number of rounds	20	-	20	-
Epoch	3	150	3	150
Batch size	2	5	2	5
Learning rate	0.0005	0.0005	0.002	0.002
SLI window size (sw)	3	3	3	3
Candidates (K)	{1, 2, 3 ..., 20}		{1, 2, 3 ..., 30}	

and from different communities, respectively. For each metric, we measured the average considering all pairs of individuals – i.e., disregarding partitioning into communities –, and compared it with the intra- and inter-community pair averages, all with 95% confidence intervals.

In the temporal aspect, for each interval t, we counted the time spent together by each pair of users in the same cell as the contact time between them. Through this metric, we expected to obtain a higher intra-community average contact time and a lower inter-community average contact time compared to the average contact time of all users.

In the spatial aspect, we computed three metrics over all pairs of mobility samples in each interval t. The *Structural SIMilarity* (SSIM) [107] compares local patterns of pixel intensities in images regarding luminance and contrast, and the *Adjusted Rand Index* (ARI) [108, 109] calculates the adjusted Rand index, which determines the hit rate of cluster label assignments for each pair of elements. As they are considered similarity metrics, we expected higher SSIM and ARI values to be obtained for intra-community samples. The *Mean Square Error* (MSE) consists of the square mean of the difference in intensity of corresponding pixels in two compared images and is used as a dissimilarity metric among users' mobility. Thus, we expected higher MSE values for inter-community samples.

Equation 4.2 presents the formulation of the SSIM metric, where μ_y and $\mu_{\bar{y}}$ are the average intensities of all pixels in images y and $\bar{y}$, respectively; σ_y and $\sigma_{\bar{y}}$ are the variances of these images; $\sigma_{y\bar{y}}$ the covariance between them; and C_1 and C_2, small constants to avoid instability when $(\mu_y^2 + \mu_{\bar{y}}^2)$ and $(\sigma_y^2 + \sigma_{\bar{y}}^2)$ are very close to zero.

$$\text{SSIM}_{(y,\bar{y})} = \frac{(2\mu_y\mu_{\bar{y}} + C_1)(2\sigma_{y\bar{y}} + C_2)}{(\mu_y^2 + \mu_{\bar{y}}^2 + C_1)(\sigma_y^2 + \sigma_{\bar{y}}^2 + C_2)} \tag{4.2}$$

Equation 4.3 presents the formulation of the ARI metric, where N corresponds to the number of samples in each interval t; V and $\bar{V}$ are the totals of subsets in two distinct partitioning (α and β, respectively) of these samples; and $s_{v\bar{v}}$, each subset

in the contingency matrix, represented by Table 4.6.

$$\text{ARI} = \frac{\binom{N}{2}\sum_{v=1}^{V}\sum_{\bar{v}=1}^{\bar{V}}\binom{s_{v\bar{v}}}{2} - [\sum_{v=1}^{V}\binom{s_v}{2}\sum_{\bar{v}=1}^{\bar{V}}\binom{s_{\bar{v}}}{2}]}{\frac{1}{2}\binom{N}{2}[\sum_{v=1}^{V}\binom{s_v}{2} + \sum_{\bar{v}=1}^{\bar{V}}\binom{s_{\bar{v}}}{2}] - [\sum_{v=1}^{V}\binom{s_v}{2}\sum_{\bar{v}=1}^{\bar{V}}\binom{s_{\bar{v}}}{2}]} \tag{4.3}$$

$\alpha \setminus \beta$	β_1	β_2	...	$\beta_{\bar{V}}$	Total
α_1	s_{11}	s_{12}	...	$s_{1\bar{V}}$	s_1
α_2	s_{21}	s_{22}	...	$s_{2\bar{V}}$	s_2
...	...	...	...	...	...
α_V	s_{V1}	s_{V2}	...	$s_{V\bar{V}}$	s_V
Total	s_1	s_2	...	$s_{\bar{V}}$	

Table 4.6: Contingency matrix

As for the MSE metric, the Equation 4.4 presents its formulation, where P corresponds to the total number of pixels compared between images y and $\bar{y}$.

$$\text{MSE}_{(y,\bar{y})} = \frac{1}{P}\sum_{i=1}^{P}(y_i - \bar{y}_i)^2 \tag{4.4}$$

Finally, to analyze these metrics, we identified and labeled as *Best* the K candidate that resulted in the highest average contact time among intra-community users. Our decision was based on the understanding that contact time is the most important metric, as it indicates the coexistence of individuals in the same cells.

Chapter 5

Results

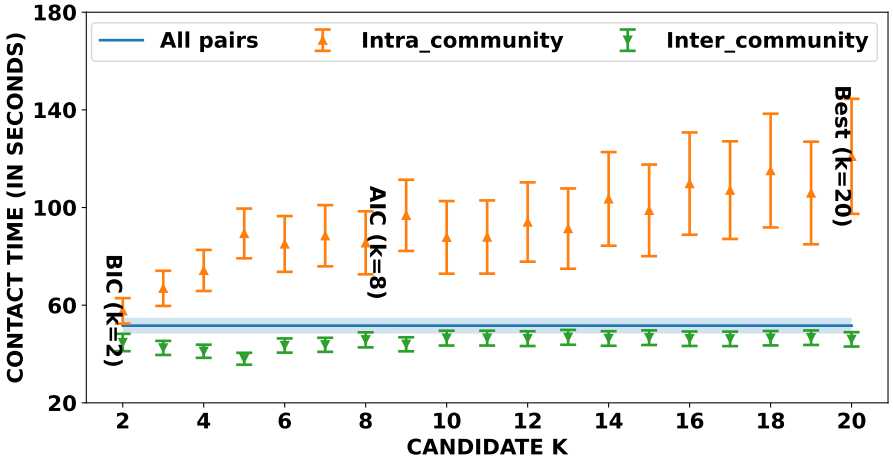
In this chapter, we present the results obtained from our experiments to answer the RQ presented in Section 4. To answer RQ1, in Section 5.1, we analyze the choice of the number of communities made by the probabilistic methods AIC and BIC. To answer RQ2, in Section 5.2, we analyze the impact of the ACC and SLI strategies on the model's results and their division into communities. In these two sections, we evaluate the results obtained from the use of the FedAvg and FedProx algorithms in specific intervals. Finally, to answer RQ3, in Section 5.3, we compare the results obtained by these two algorithms against those of the centralized training approach over several intervals.

5.1 The implications of using AIC and BIC to define K

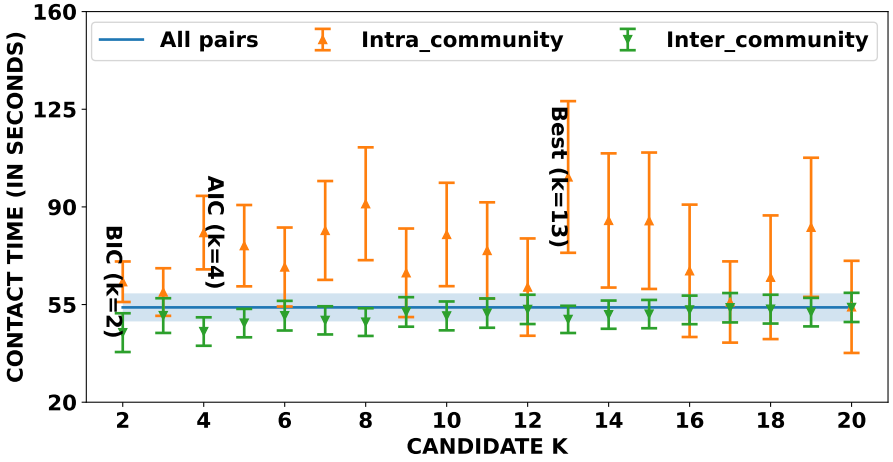
For most intervals in each scenario, we observed divergent choices of K between the probabilistic methods. Since the SLI window has size $sw = 3$, in the intervals t_0, t_1 and t_2, the ACC and SLI strategies use the same training samples. Thus, the interval t_2 is the last one in which the results of these strategies are equal. Figures 5.1 and 5.2 show, respectively, the FedAvg and FedProx average contact times obtained in the SFC, RT and NGSIM scenarios during the interval t_2 for all pairs of individuals and for intra- and inter-community individuals. We noticed a fluctuation in the average intra-community contact time as the K candidate increases in the three scenarios. Quite similar oscillation patterns can be observed between the candidates in the FedAvg and FedProx SSIM (Figures 5.3 and 5.4, respectively), and MSE (Figures 5.5 and 5.6, respectively), both referring to the interval t_2. These results strongly suggest that a random choice of K – or even keeping it fixed over different intervals – can produce community formations in which the average intra-community contact time may be negatively affected. Therefore, this confirms that choosing a candidate

requires a selection criterion.

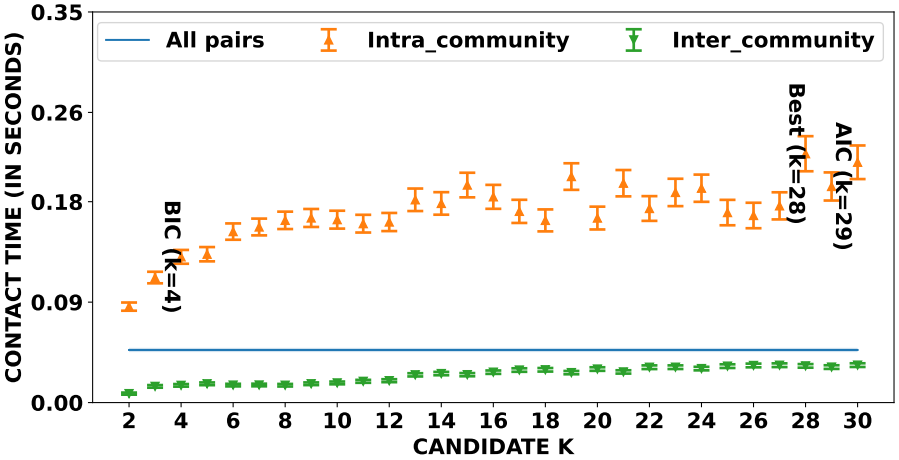
Since mobility samples – and their respective encoded representations – express



(a) Contact time for SFC



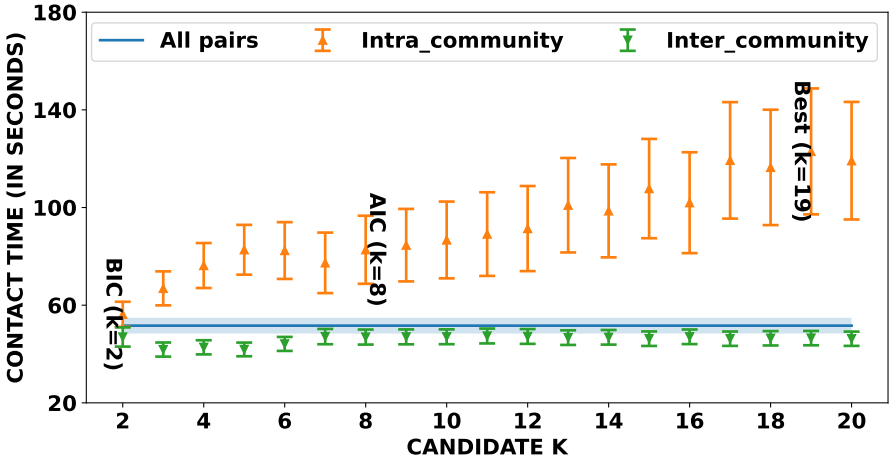
(b) Contact time for RT



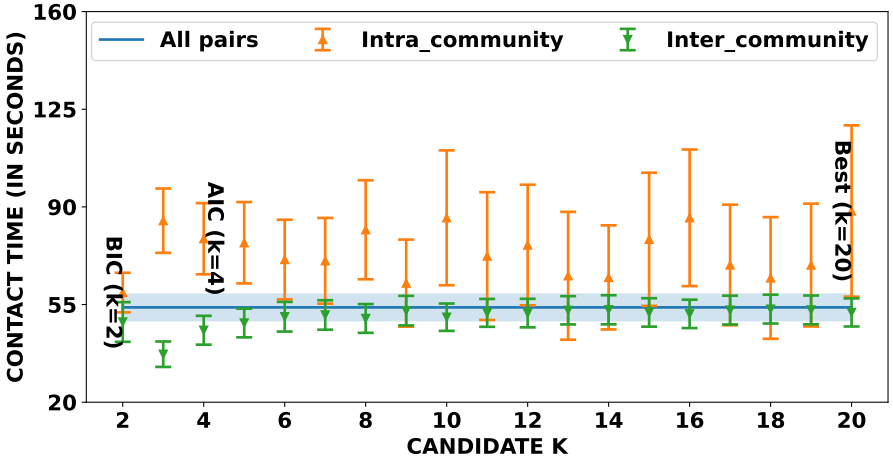
(c) Contact time for NGSIM

Figure 5.1: FedAvg average contact time at interval t_2 for SFC, RT and NGSIM scenarios.

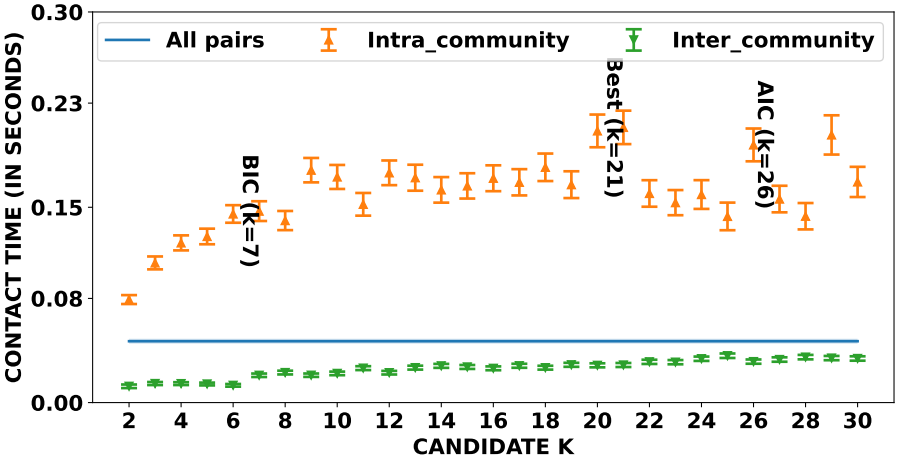
more the spatial than the temporal aspect of the individuals' behavior, we can say that similarity and dissimilarity are direct metrics, while contact time is an indirect



(a) Contact time for SFC



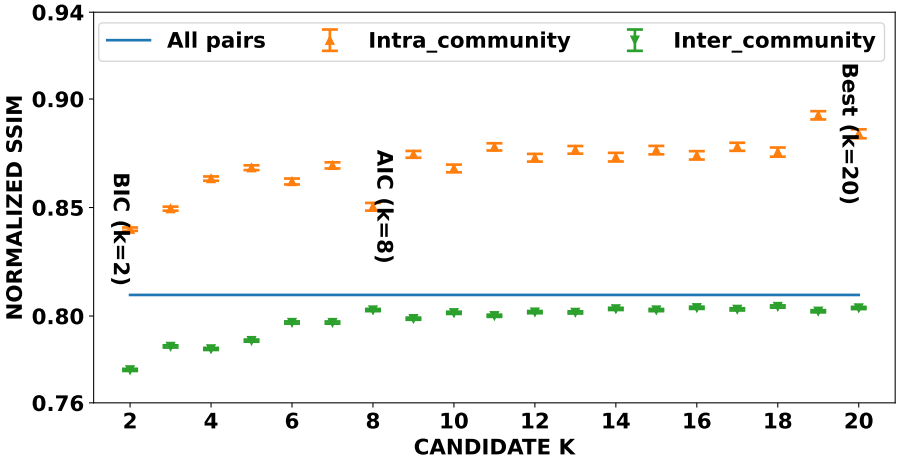
(b) Contact time for RT



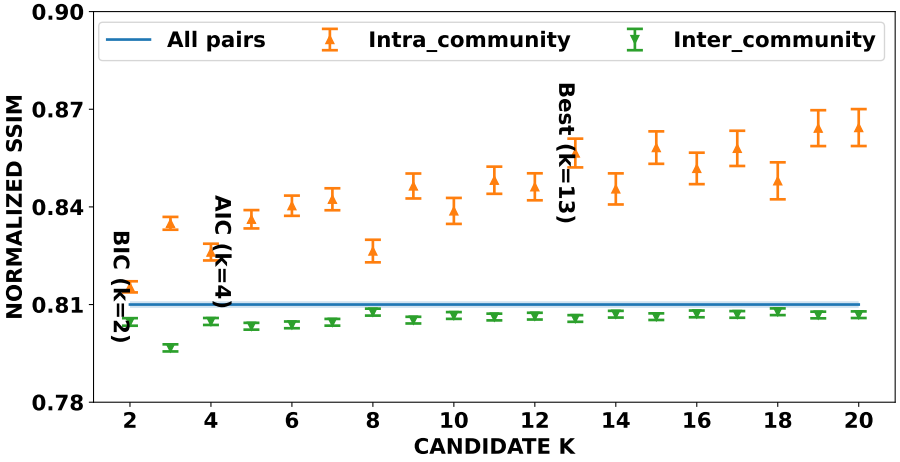
(c) Contact time for NGSIM

Figure 5.2: FedProx average contact time at interval t_2 for SFC, RT and NGSIM scenarios.

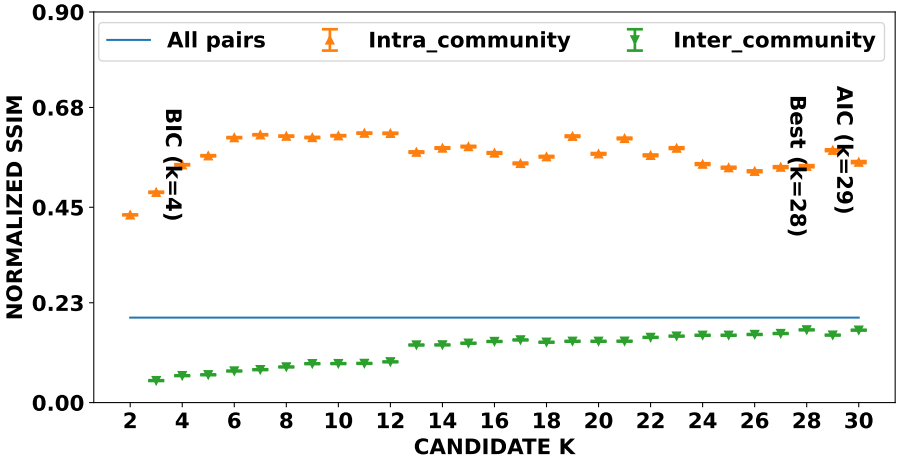
evaluation metric. This is why a temporal resolution setting is so important because even if two individuals take the same paths within the same time interval, if the



(a) SSIM for SFC



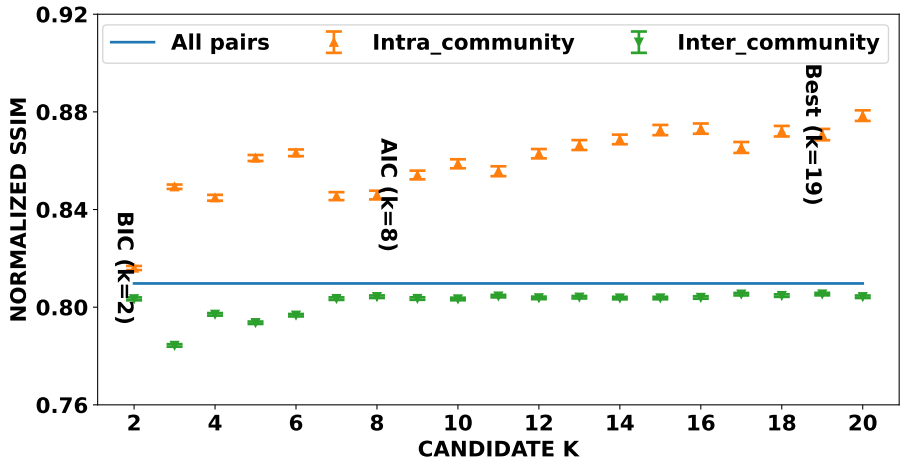
(b) SSIM for RT



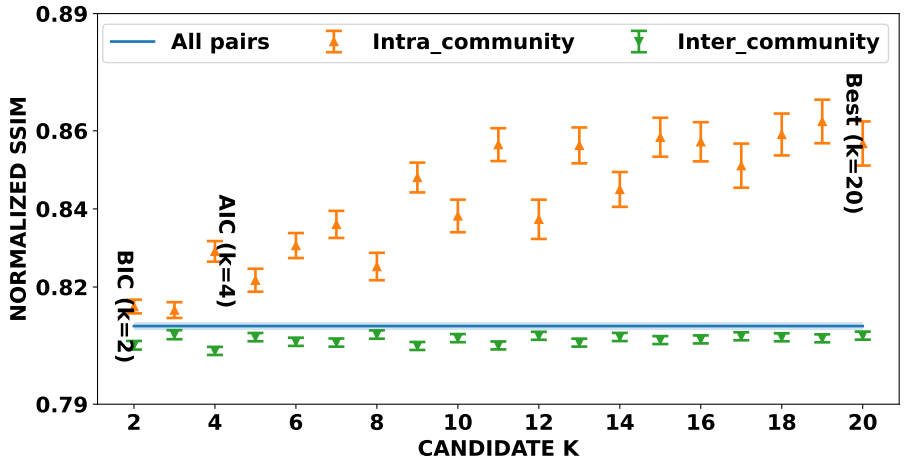
(c) SSIM for NGSIM

Figure 5.3: FedAvg average similarity (SSIM) metric at interval t_2 for SFC, RT and NGSIM scenarios.

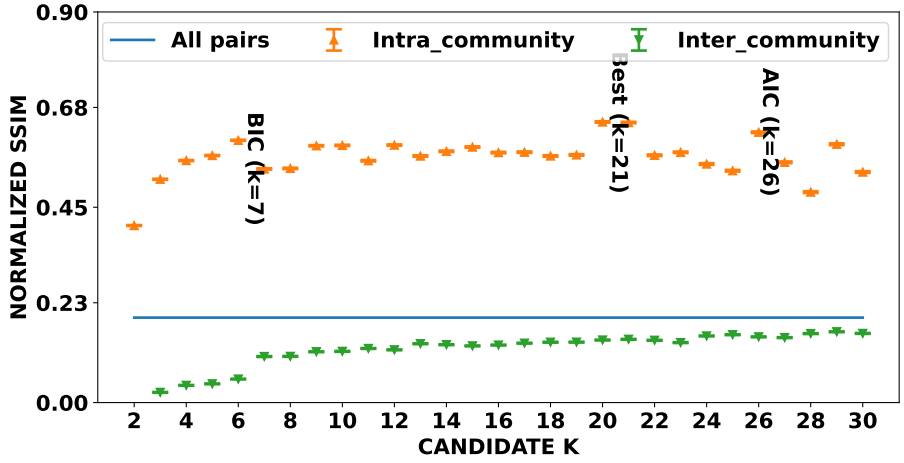
interval is large enough that they do not coexist in the same cells, their contact time will be zero, even though the mobility samples are identical, which would produce



(a) SSIM for SFC



(b) SSIM for RT

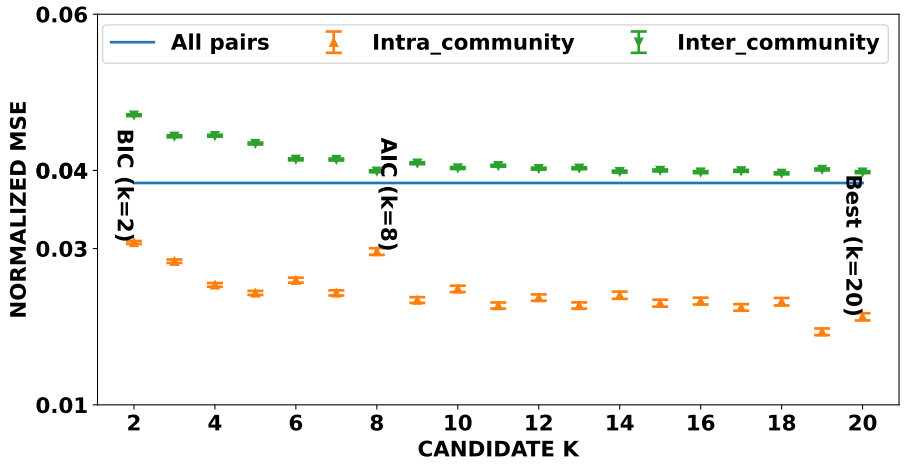


(c) SSIM for NGSIM

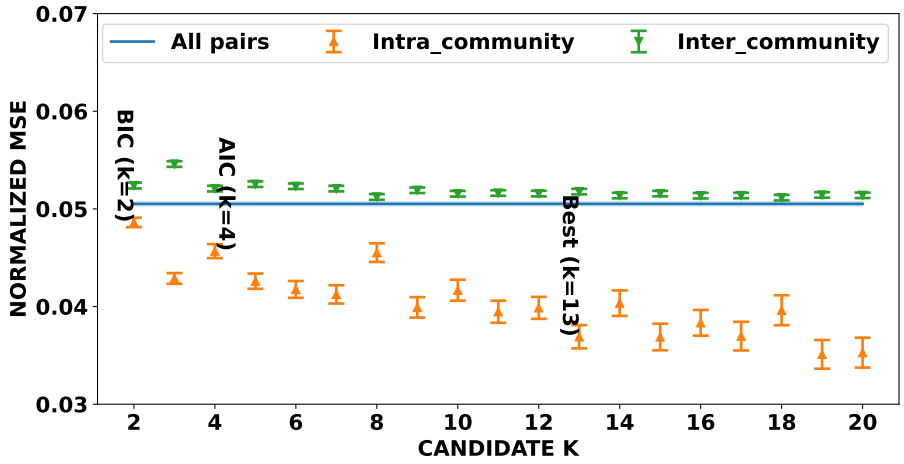
Figure 5.4: FedProx average similarity (SSIM) metric at interval t_2 for SFC, RT and NGSIM scenarios.

excellent similarity and dissimilarity results.

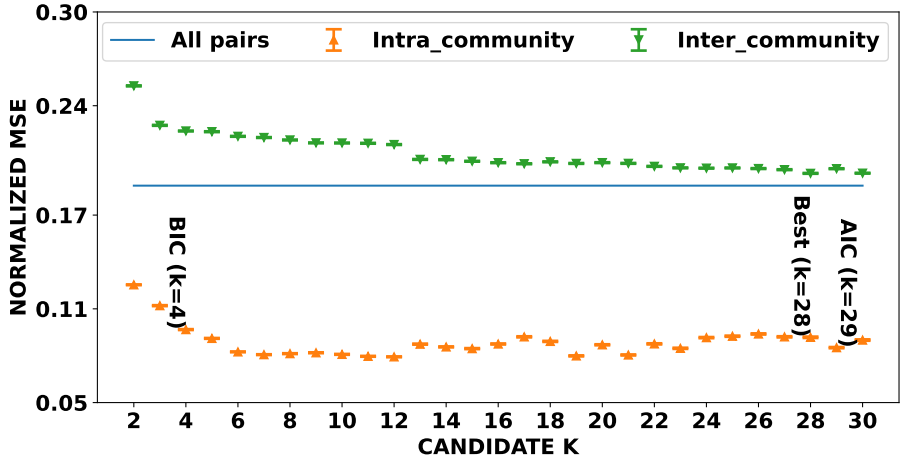
Unlike the NGSIM scenario, the SFC and RT scenarios cover a larger spatial



(a) MSE for SFC



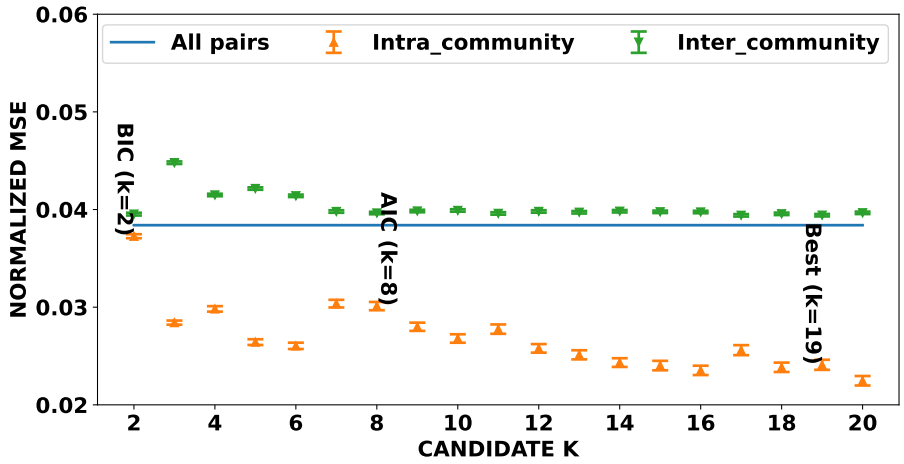
(b) MSE for RT



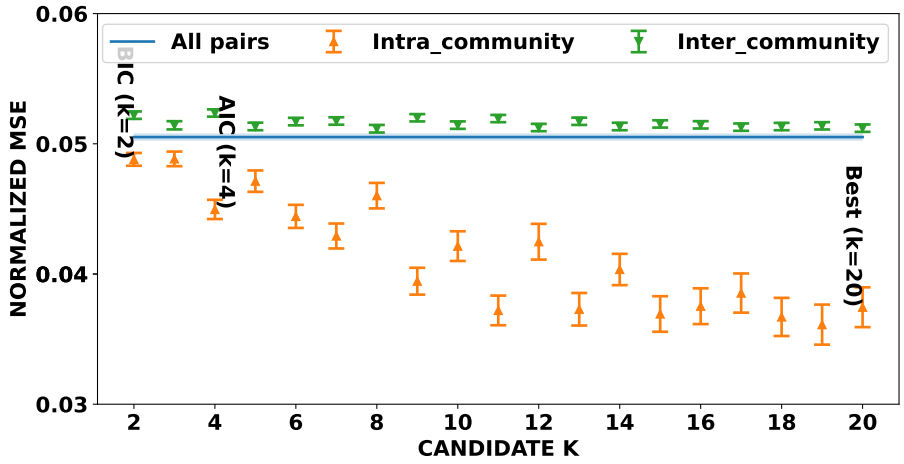
(c) MSE for NGSIM

Figure 5.5: FedAvg average dissimilarity (MSE) metric at interval t_2 for SFC, RT and NGSIM scenarios.

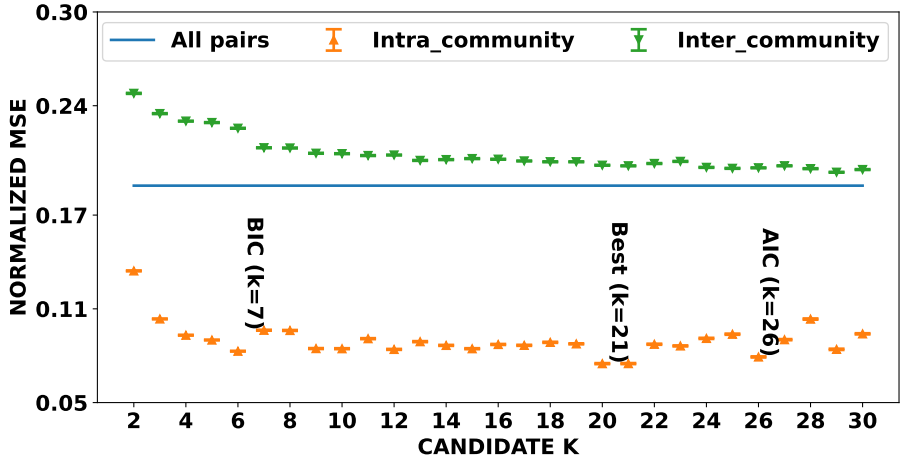
area, and have smaller spatio-temporal resolutions (larger cells and intervals). They also have a smaller number of users, which directly influences their density. As a



(a) MSE for SFC



(b) MSE for RT



(c) MSE for NGSIM

Figure 5.6: FedProx average dissimilarity (MSE) metric at interval t_2 for SFC, RT and NGSIM scenarios.

result, small changes in community composition have a significant impact on the evaluation metrics in these scenarios. This especially explains the result for contact time in the RT scenario. That is, this scenario presents the lowest density of users – who may not even be present on the map at the same time. Therefore, two or more individuals grouped in the same community may present the same movement pattern, but at different times within an interval. This implies zero contact time between them, even if there is spatial similarity. As a result, we see that certain K candidates produce intra-community contact time lower than the average for all users in the RT scenario.

In theory, choosing higher K values would result in better-defined communities, i.e. where users share a very high degree of behavioral similarity, as expressed by the evaluation metrics. However, due to the characteristics and configurations of the scenarios, this assumption does not always prove to be true, which can be verified by the oscillation observed in the results as K increases.

Despite this, we observed that the AIC method tends to choose higher K values compared to the BIC, as shown in the Figures 5.1, 5.3 and 5.5. On these occasions, we also found that AIC had better results than BIC. Although both results are not always as close as those obtained by the *Best* candidate, they still improve intra- and inter-community metrics.

Thus, in response to RQ1, we found that the AIC method outperformed the BIC, although both met the expectations of the evaluation metrics. Although the results of all the metrics may be some distance from their best value, the probabilistic methods provided safer choices of K, ensuring that the intra- and inter-community metrics of similarity and dissimilarity are distant from the average of all pairs.

5.2 The impacts of ACC and SLI strategies for FL-based training and communities' composition

Since the SLI window size is $sw = 3$, the results of this strategy and the ACC ones are the same in windows $[0, 1)$, $[0, 2)$ and $[0, 3)$. As the SLI window slides and training datasets become different for each strategy, from the interval t_3 onwards, the results start to diverge in windows $[0, 4)$ (ACC) and $[1, 4)$ (SLI). Thus, Figures 5.7, 5.8, 5.9 present the loss curves of the FedAvg and FedProx global models in later windows, so that the impact of different amounts of samples on model training can be observed.

In all scenarios and strategies, although subtle, the downward trend remains for both algorithms, which indicates that the model is learning. We observe that the ACC and SLI window-strategies generated loss curves with very small distances between training and testing. Still, in the SFC and RT scenarios, the ACC strategy

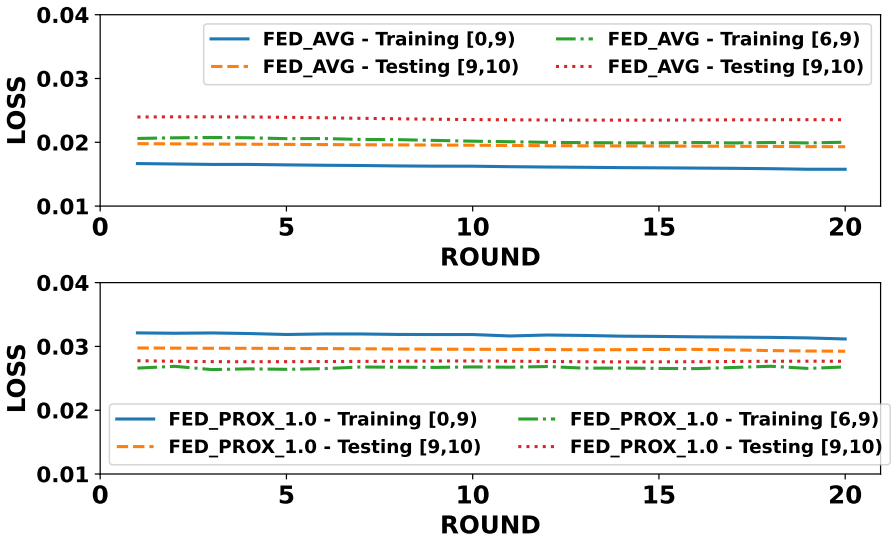


Figure 5.7: Losses of the FedAvg and FedProx global models in the SFC scenario after training using the windows [0, 9) and [6, 9), i.e., ACC and SLI, respectively. Samples from the interval t_9 were used for testing.

had the lowest loss in both curves when using the FedAvg algorithm, while the SLI strategy had the best result when using FedProx. In contrast, in the NGSIM scenario, the ACC strategy performed better with both algorithms, although with FedProx the difference between ACC and SLI is practically negligible.

Because the SFC and RT scenarios are more extensive - both spatially and temporally - and therefore more complex, the largest number of samples used by ACC help to improve the model. In this way, the model undergoes more updates and,

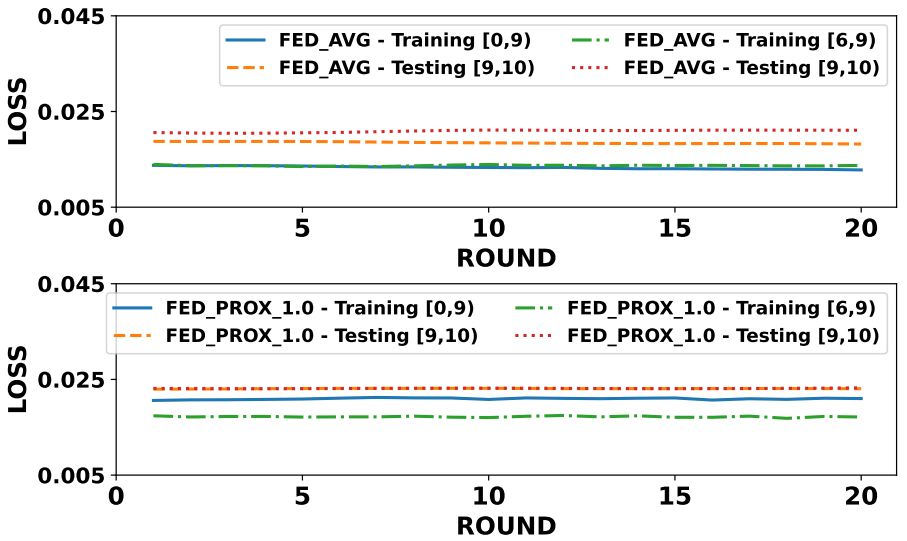


Figure 5.8: Losses of the FedAvg and FedProx global models in the RT scenario after training using the windows [0, 9) and [6, 9), i.e., ACC and SLI, respectively. Samples from the interval t_9 were used for testing.

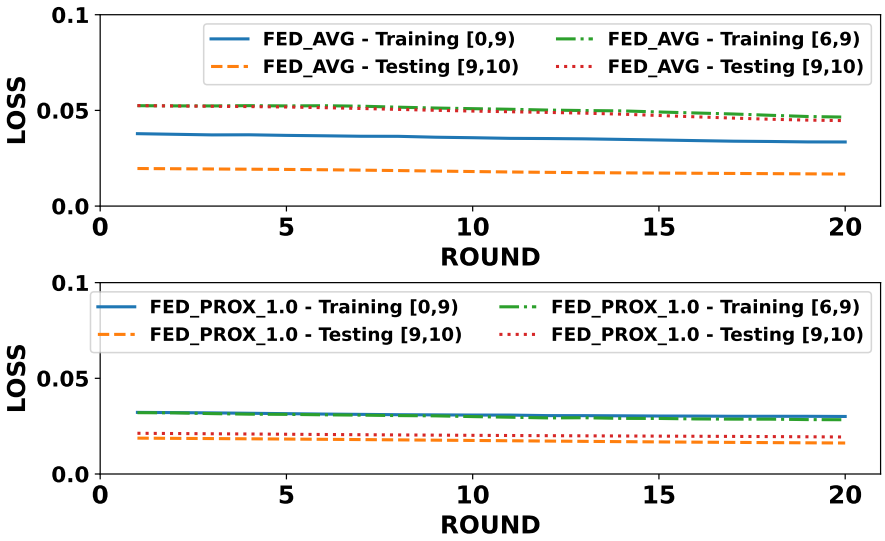
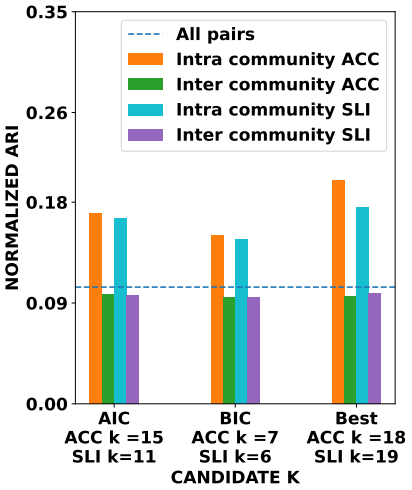


Figure 5.9: Losses of the FedAvg and FedProx global models in the NGSIM scenario after training using the windows [0, 9) and [6, 9), i.e., ACC and SLI, respectively. Samples from the interval t_9 were used for testing.

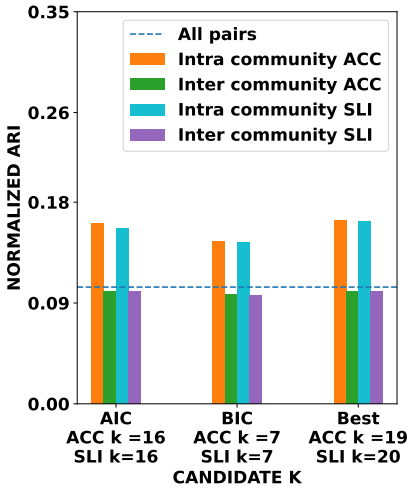
consequently, its parameters are better adjusted. Regarding the better performance of the SLI strategy when using FedProx, the smaller number of samples subjects the model to fewer updates during local training. This implies a reduction in deviations in relation to the global model, which favors learning. Despite this, we observe that the FedAvg algorithm presented lower losses than FedProx in these scenarios. In both cases, we verified that the test curve remains above the training curve. However, as the downward trend continues throughout the rounds, it is not an indication of overfitting.

In turn, in the NGSIM scenario, both ACC and SLI showed test curves equal or below the training curve, indicating the model's high generalization capacity in this scenario with both window strategies. Although the ACC strategy also performs better with FedProx, we observe a significant improvement in the loss generated by SLI with FedProx compared to FedAvg, which reiterates our analysis. Furthermore, although the loss values obtained in the NGSIM scenario can be considered small, they were higher compared to the previous scenarios. However, through empirical analysis of the sample reconstructions, we observed that the ACC and SLI models achieved satisfactory quality. Thus, we confirmed that both ACC and SLI produce models with low reconstruction loss, which indicates that the resulting encoded representations are capable of expressing the individuals' behavioral patterns.

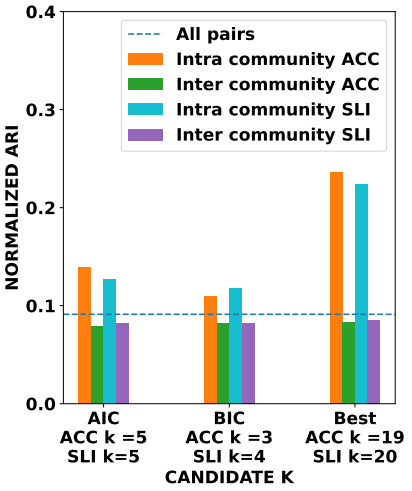
To compare the strategies' performance, we analyze the average ARI in the t_8 interval, which provides the last mobility sample of the training windows [0, 9) and [6, 9). Thus, Figure 5.10 shows the average of this metric when combining the window strategies with the K choices made by the probabilistic methods.



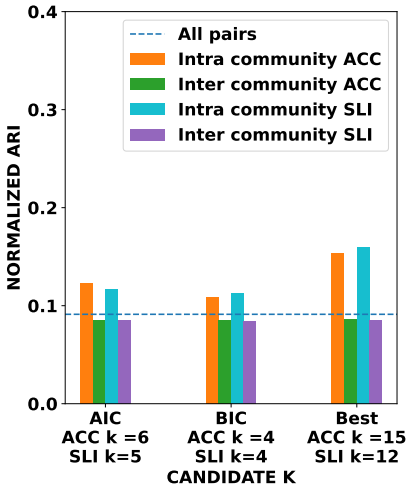
(a) FedAvg SFC at interval t_8



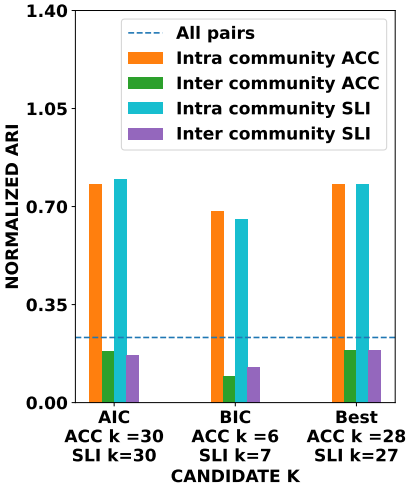
(b) FedProx SFC at interval t_8



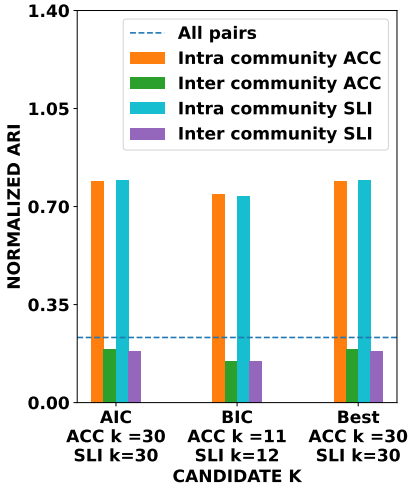
(c) FedAvg RT at interval t_8



(d) FedProx RT at interval t_8



(e) FedAvg NGSIM at interval t_8



(f) FedProx NGSIM at interval t_8

Figure 5.10: Average ARI for the SFC, RT and NGSIM scenarios at interval t_8 – after training in windows $[0, 9)$ (ACC) and $[6, 9)$ (SLI).

In the SFC scenario, we confirmed the better performance of the ACC strategy over SLI, given that in all cases, even by a tiny difference, the former resulted in a higher ARI. Regarding the FedProx algorithm, it would be natural for SLI to result in a higher average ARI, especially in cases where the same K is chosen, since this strategy obtained lower training and testing losses compared to ACC. However, we observe a certain equity between the results. This indicates that even with the lower learning loss, the predictive capacity of the model trained with SLI is very similar to that of ACC when using FedProx. This is justified by the penalty that this algorithm gives to deviations of local models in relation to the global model. That is, when using ACC, it is common for some users to have many more samples than others, which would lead them to make more updates to the local model. However, when the FedProx penalty is applied, the impact of these updates is softened. On the other hand, when using SLI, a limit of samples for training is imposed, which reduces the impact of the FedProx penalty on the trained model while balancing its result with that of the ACC model.

In the RT scenario, in line with the learning loss results, we see that the ACC strategy performed better with FedAvg, while SLI performed better with FedProx. In this case, we highlight the combinations that chose the same K, i.e., FedAvg/AIC and FedProx/BIC. These results clearly show the gain of each strategy. When comparing the results of FedAvg and FedProx, we observed that the former obtained better performance of the metric in the corresponding combinations, even when the K is the same. As this is a scenario with a significantly smaller number of users and, therefore, samples, the use of FedAvg helps the model to learn, by facilitating its improvement and allowing the full participation of all available users. That is, FedAvg performs aggregation in a simple way, without explicit penalties to deal with the heterogeneity of local data. Thus, when using ACC, the algorithm takes advantage of all available samples, offering a larger and more diverse database for local training, which can benefit the learning in scenarios with few users. In contrast, the FedProx algorithm benefits from the use of the SLI strategy, with which only the most recent samples are used, and local heterogeneity is reduced (local data are more homogeneous, as they reflect more aligned behavioral temporal patterns). In this case, the FedProx penalty term is activated less frequently, allowing local updates to contribute more freely to the global model.

Finally, in the NGSIM scenario, we found very close results between ACC and SLI when the FedProx algorithm was used. This result is consistent with that obtained in the loss function curves. We attribute this result to two main factors. First, to the characteristics of the scenario. That is, since it is a single high-speed highway, the travel patterns among all users tend to be quite similar, regardless of whether they are more recent samples or not. Furthermore, given that this scenario has very few gaps

in mobility records, the number of samples among all users tends to be practically the same, which balances their participation. Thus, the older samples used by ACC in this scenario end up having reduced impact on the model when compared to the more recent ones, while the contribution of users tends to be practically the same in both strategies. This is reinforced by the results of the strategies when FedAvg is used. In all combinations with this algorithm, especially the *Best* candidate, we observe values very close to those obtained with FedProx. This shows that in this scenario, where the behavior patterns of all users, as well as their number of local samples, tend to be practically the same, the FedAvg and FedProx models achieve quite similar learning capacity.

To corroborate our analysis, Tables 5.1, 5.2 and 5.3 show the average intra-community ARI when ACC and SLI produce the same number of communities in the SFC, RT and NGSIM scenarios, respectively. The values presented in these tables, especially those related to the SFC and RT scenarios, confirm that FedAvg benefits from the larger volume of data (ACC), even if heterogeneous. This is because the lack of regularization makes it depend on diversity and quantity of data to ensure generalization. In turn, using only recent samples (SLI) reduces diversity, which impacts this capacity. In turn, FedProx, when using all local samples, faces greater heterogeneity, which triggers the penalty more frequently, limiting the impact of local updates. Finally, by using only more recent samples, this algorithm applies less penalty, allowing local updates to contribute more significantly to the global model. As for NGSIM, with its large number of participants, the influence of each user on the global model is reduced, which smooths out the impacts of individual variations and facilitates the convergence of the global model. In addition, the similarity between user datasets reduces global heterogeneity. Thus, both FedAvg and FedProx benefit from the SLI strategy, which filters out the most recent data that has the most impact, resulting in more focused and efficient updates to the global model.

These results reinforce the ability of the FL-based FCAE model to extract spatio-temporal mobility features to identify communities. In this context, using the ACC and SLI window-strategies showed that training performed with different amounts of samples directly impacts the communities' composition. We observed that in scenarios with greater spatial extent and lower temporal resolution (like SFC and

Table 5.1: Average intra-community ARI for each window-strategy when both produce the same number of communities in the SFC scenario.

Strategy	FedAvg						FedProx			
	k=6	k=7	k=11	k=15	k=18	k=19	k=7	k=16	k=19	k=20
ACC	0.1498	0.1504	0.1545	0.1697	0.1991	0.1843	0.1448	0.1608	0.1639	0.1628
SLI	0.1470	0.1506	0.1657	0.1691	0.1774	0.1753	0.1444	0.1563	0.1641	0.1632

Table 5.2: Average intra-community ARI for each window-strategy when both produce the same number of communities in the RT scenario.

Strategy	FedAvg					FedProx				
	k=3	k=4	k=5	k=19	k=20	k=4	k=5	k=6	k=12	k=15
ACC	0.1096	0.1325	0.1390	0.2360	0.2113	0.1087	0.1135	0.1233	0.1388	0.1537
SLI	0.1088	0.1181	0.1270	0.2156	0.2241	0.1123	0.1166	0.1195	0.1592	0.1708

RT), the ACC strategy was more efficient than the SLI strategy. That, however, does not invalidate the SLI strategy, as it performed well in the scenario with a smaller spatial extent and higher temporal resolution. With that, we answer RQ2.

5.3 Comparison between the FL-based and centralized approaches

Since traditional (centralized) learning methods usually generate better-trained models compared to distributed approaches [110], we investigated the performance of that learning method compared to FL in identifying communities. For this, we analyzed the communities generated by the candidate *Best* in 42 subsequent intervals in the SFC, 30 in RT scenarios and 10 in the NGSIM scenario. Table 5.4 presents the *Best K* of each interval for each combination of approach and window strategy for each of the scenarios. The figures in this section show the averages of the contact time and SSIM metrics and their respective 95% confidence intervals, all interpolated from one interval to the next.

The Figure 5.11(a) presents the contact time metric for the SFC scenario. As expected, centralized training resulted in communities that better met the intra- and inter-community contact time metric expectations, which can be seen by the distance between the confidence intervals of their measurements and the average of all pairs. However, we also observed that the results obtained by all training approaches (centralized and the two FL-based) often overlap. These overlaps indicate that, in general, there is not a significant dissonance among the results of these approaches in relation to what is expected for this metric.

Still on the contact time metric of the SFC scenario, it is interesting to observe

Table 5.3: Average intra-community NGSIM for each window-strategy when both produce the same number of communities in the NGSIM scenario.

Strategy	FedAvg					FedProx		
	k=6	k=7	k=27	k=28	k=30	k=11	k=12	k=30
ACC	0.6824	0.6913	0.7796	0.7794	0.7797	0.7449	0.7528	0.7921
SLI	0.6180	0.6537	0.7803	0.7919	0.7970	0.7488	0.7361	0.7957

the peaks in the average for all nodes (solid green curve), which repeat every 3 intervals (that is, every 12 hours). In addition, we observe that the four peaks between intervals t_{27} and t_{35} are slightly different from the others. We believe that this occurs because these intervals correspond to a weekend (May 31 and June 1, 2008) and, therefore, mobility and contact patterns between users are different from those on weekdays. When analyzing the intra- and inter-community contact time obtained from our framework, we observed that these behaviors are strictly respected. Furthermore, we found that grouping users who have similar behavioral patterns significantly increases the contact time between them.

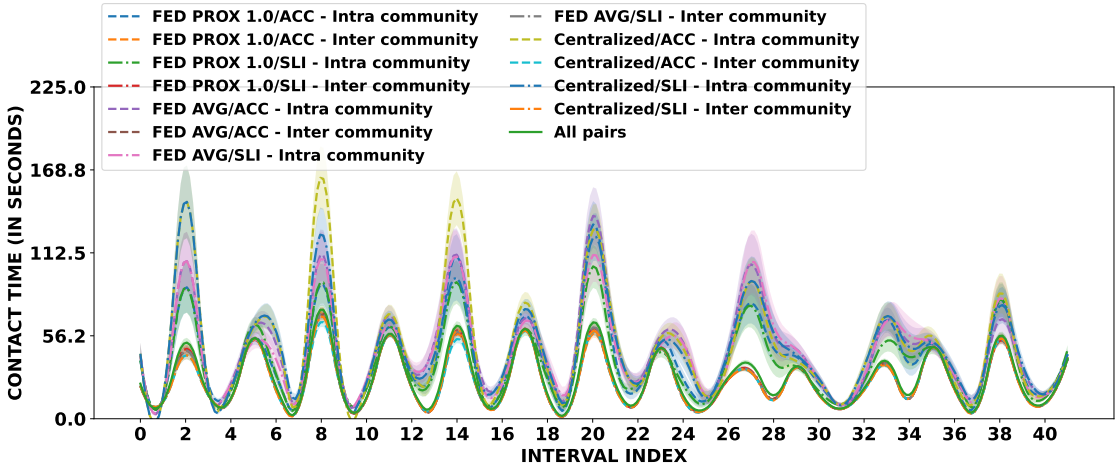
In Figure 5.11(b) we observed that all approaches also produce close results regarding the spatial similarity metric. Unlike what happens in contact time, where in most intervals we see overlapping results, in the SSIM metric we see the predominance of gains from the FL-based approach that uses FevAvg. This gain is observed when both the ACC and SLI strategies are used. This result highlights how the use of an FL-based approach can, as much as the centralized one, refine the prediction capacity of the FCAE model for extracting latent spatio-temporal features in user mobility samples. Furthermore, we verify again that the patterns in the curves repeat themselves regularly, now every 6 intervals (24 hours), and also that between

Table 5.4: *Best K* of each interval for the SFC, RT and NGSIM scenarios.

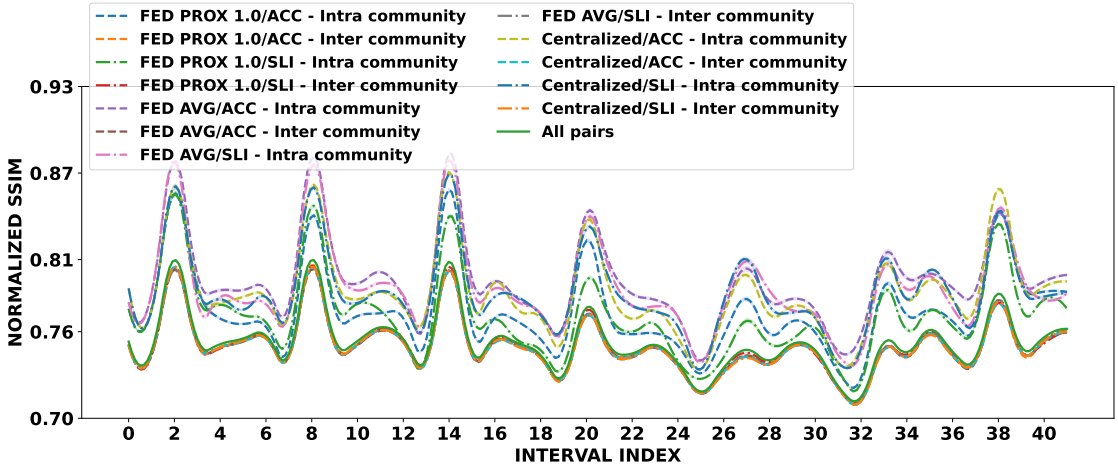
Interval	San Francisco Cabs						Roma Taxi						NGSIM					
	Centralized		FedAvg		FedProx		Centralized		FedAvg		FedProx		Centralized		FedAvg		FedProx	
	ACC	SLI	ACC	SLI	ACC	SLI	ACC	SLI	ACC	SLI	ACC	SLI	ACC	SLI	ACC	SLI	ACC	SLI
0	15	15	17	17	11	11	16	16	18	18	15	15	30	30	28	28	28	28
1	11	11	19	19	19	19	7	7	18	18	5	5	28	28	24	24	23	23
2	14	14	20	20	19	19	11	11	13	13	20	20	23	23	28	28	21	21
3	18	17	20	20	14	16	16	11	19	12	20	8	27	26	28	29	26	30
4	17	20	20	20	20	7	5	3	16	18	12	9	25	30	22	29	23	29
5	18	14	18	18	15	13	4	17	15	17	19	18	21	21	29	28	29	29
6	19	16	19	19	16	19	7	4	18	10	13	19	22	17	24	29	27	29
7	20	19	11	11	20	20	6	3	19	17	20	12	28	27	29	28	29	26
8	15	20	18	18	19	20	8	16	19	20	15	12	28	28	28	27	30	30
9	20	20	8	8	16	17	12	12	16	17	14	16	30	22	30	28	28	19
10	14	17	20	20	19	20	9	20	15	16	13	20						
11	17	20	15	15	18	19	17	17	18	4	15	12						
12	20	17	15	15	20	19	17	16	20	20	9	13						
13	20	20	20	20	20	20	6	2	9	3	16	4						
14	14	14	20	20	18	19	18	19	20	20	19	18						
15	17	18	17	17	18	20	14	18	15	10	5	20						
16	17	16	19	19	20	19	20	13	18	12	19	10						
17	14	10	11	11	20	18	20	20	19	16	18	13						
18	18	17	19	19	19	20	3	9	14	9	9	19						
19	20	18	18	18	20	20	2	11	2	2	9	4						
20	20	20	20	20	20	19	18	17	18	19	17	12						
21	18	20	16	16	19	19	19	13	14	5	13	15						
22	18	17	20	20	19	19	13	16	20	9	16	18						
23	20	18	20	20	20	17	19	14	11	14	19	15						
24	20	20	20	20	17	20	13	17	14	19	18	8						
25	17	18	19	19	19	20	4	2	2	6	17	8						
26	17	19	16	16	17	18	19	18	7	4	11	20						
27	19	20	20	20	20	20	19	12	14	20	19	11						
28	18	18	18	18	20	20	14	19	15	14	18	8						
29	19	20	19	19	18	14	16	12	20	13	18	19						
30	16	20	19	19	20	13												
31	19	19	14	14	18	17												
32	16	17	20	20	15	16												
33	16	20	18	18	19	20												
34	20	19	18	18	19	20												
35	19	19	19	19	20	19												
36	20	20	20	20	20	15												
37	20	14	20	20	17	18												
38	16	18	19	19	20	18												
39	18	16	19	19	18	20												
40	14	18	18	18	20	9												
41	19	18	15	15	17	14												

intervals t_{27} and t_{35} the pattern is different. Since the intra- and inter-community SSIM – for all approaches and strategies – also follow these patterns, but with a considerable improvement over the average for all nodes, we confirm the similarity among the results of the approaches in this scenario.

In turn, Figure 5.12(a) presents the contact time metric for the RT scenario. In this scenario, we do not observe well-defined patterns that repeat themselves in the average contact time for all pairs of users (solid green curve). In fact, we consider this scenario to be quite challenging since it has a low number of users, which sometimes varies abruptly from one interval to the next, as shown in Table 5.5. This feature allows us to understand the result presented in this figure, especially if we consider the result obtained in the SSIM metric (Figure 5.12(b)). The joint analysis of the contact time and the similarity metric in the RT scenario confirms our assumption

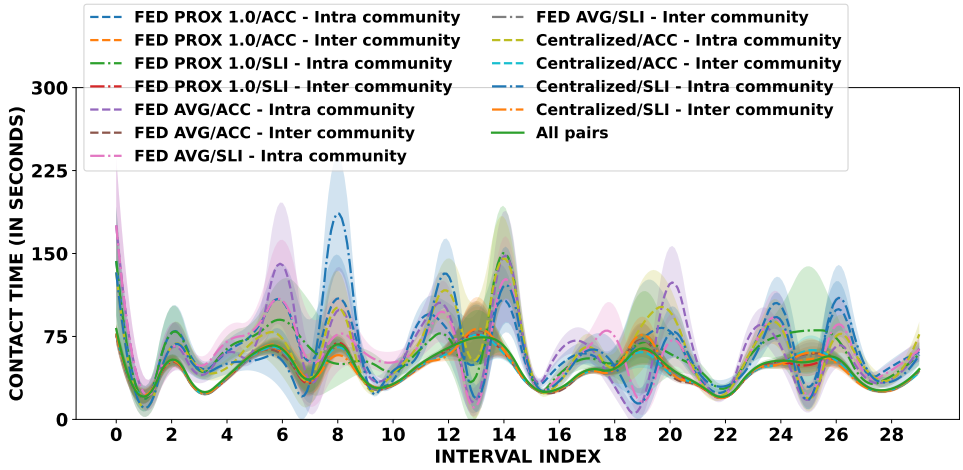


(a) Contact time

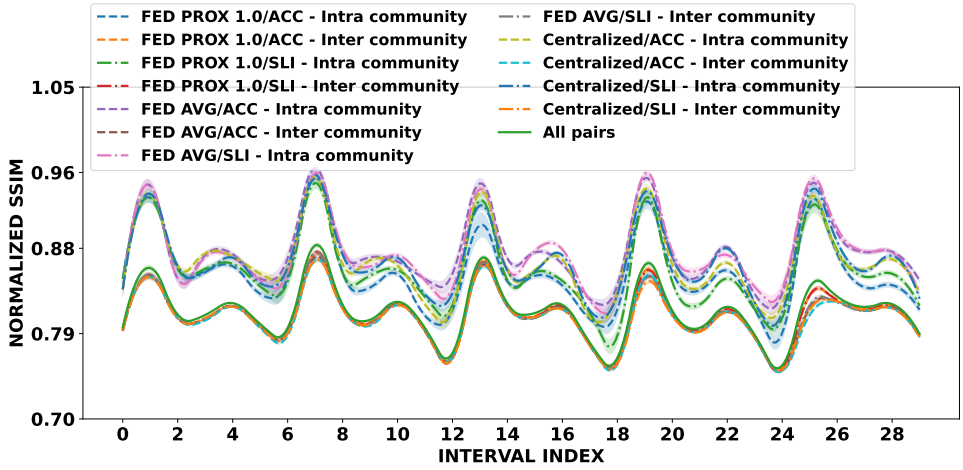


(b) SSIM

Figure 5.11: Contact time and SSIM metrics for the SFC scenario using FL-based (FedAvg and FedProx) and centralized approaches with the ACC and SLI window-strategies.



(a) Contact time



(b) SSIM

Figure 5.12: Contact time and SSIM metrics for the RT scenario using FL-based (FedAvg and FedProx) and centralized approaches with the ACC and SLI window-strategies.

that the samples and their respective coded representations express more the spatial than the temporal aspect of the mobility of individuals.

This becomes clear when we notice not only the occurrence of these patterns that repeat themselves in the SSIM metric, but also the way in which the intra- and inter-community SSIM respect these patterns. This occurs precisely because, even

Table 5.5: Number of nodes in each interval in the RT scenario.

Interval	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Nodes	65	50	117	183	168	139	71	48	114	172	177	146	80	46	118

Interval	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
Nodes	179	180	140	80	51	122	166	182	149	83	55	121	172	181	148

though there are few users at certain intervals, there are similar behavioral patterns among them. However, this does not imply that they necessarily occur at the same or close moments. For this reason, the intra- and inter-community contact time in this scenario is so irregular.

In Table 5.5 we highlight the intervals in which the number of users reduces drastically. When we check Table 5.4, we see that the *Best K* value drops dramatically in these intervals – especially where the number of nodes is below 60 – compared to the others. Looking at Figure 5.12(a), we see that exactly in these intervals, the framework fails to increase the intra-community contact time. In fact, this contact time ends up being below the average for all pairs of users. However, this failure is not repeated in the similarity metric. This finding confirms our premise and reinforces that, to generate better intra-community contact times, our framework is dependent on the granularity of the temporal resolution in the pre-processing step. As in the previous case, we observe in this scenario that there is an alternation between the approaches that produce greater intra-community contact time. However, in the SSIM metric, again, we see the predominance of gains from the approach based on FL, FedAvg. This reinforces once again the advantages of using a federated approach.

Finally, for the NGSIM scenario, Figure 5.13 presents the contact time and the SSIM metric. As in the previous scenarios, we observe alternating gains between the approaches in both metrics but a predominance of the FL-based approach in the SSIM metric. This reiterates the quality of the model trained via FL compared to a traditional approach.

Unlike the SFC scenario, we do not observe patterns in the curves of both metrics. This is because the NGSIM scenario covers a short period of time. Furthermore, NGSIM records the mobility of a huge number of users, which maintains, in each interval, a reasonable quantity of individuals moving. With a temporal resolution that accommodates a high number of nodes within the same cells simultaneously, the contact time between them increases. Thus, unlike the RT scenario, we see in Figure 5.13(a) that the intra-community contact time is higher than the average for all nodes in all intervals[1], although the scale is smaller.

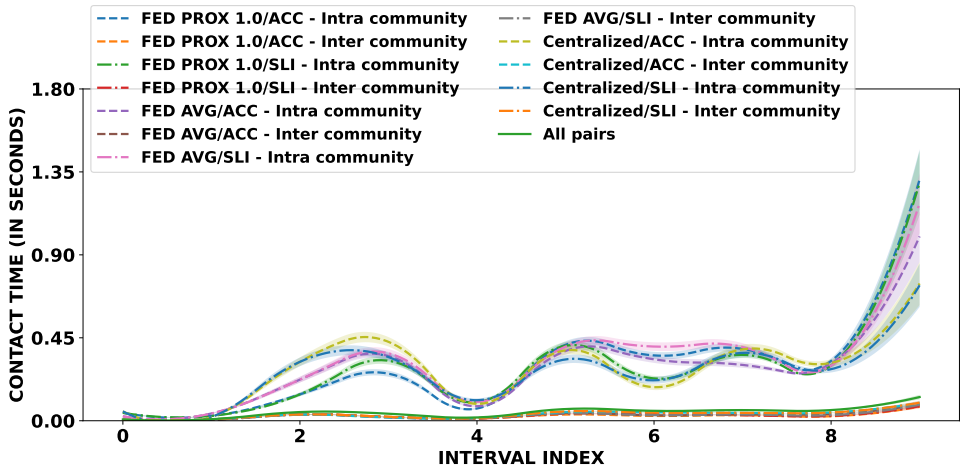
Similar behavior can be seen in Figure 5.13(b), referring to the SSIM metric for the NGSIM scenario. Again, the results obtained by each combination of approach and strategy are very similar in most intervals. Furthermore, in this scenario, the methodology's ability to distance the intra-community similarity curves from the average for all pairs of nodes is more evident than in the previous ones. The choices of the same or nearby K in some intervals keep the approaches' curves very close to each other, for example, the intervals t_0 and t_8. This reinforces that the use of an

[1]In intervals t_0 and t_1, the minimum difference between the intra-community and all-pairs average contact time is 0.03 and 0.04, respectively.

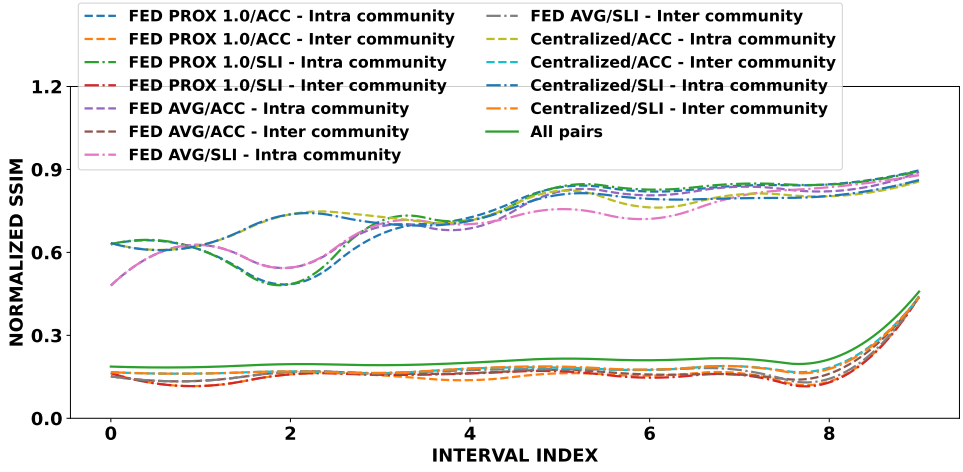
FL-based solution is not far from a centralized one in the this scenario.

It is interesting to note from the NGSIM scenario that its temporal resolution is quite different compared to the SFC and RT scenarios. Although the spatial resolution is also quite different, with NGSIM we were able to see what happens to the results when the size of the interval t is reduced and the communities are updated more frequently. With the results of this scenario, we saw that the contact time and the similarity metric undergo smoother changes from one interval to the next and, above all, among the combinations of approach and window-strategy.

The results presented in this subsection for the SFC, RT and NGSIM scenarios demonstrate that the FL-based approach achieves results very close to those obtained by the centralized approach for the contact time and SSIM metrics in community identification. Thus, answering RQ3, We found that there is no significant difference



(a) Contact time



(b) SSIM

Figure 5.13: Contact time and SSIM metrics for the NGSIM scenario using FL-based (FedAvg and FedProx) and centralized approaches with the ACC and SLI window-strategies.

between the results obtained by both approaches, even with the window-strategies, as we observed from the similar trends and multiple overlapping curves.

5.4 Discussion

In this section we present a brief discussion of how time influences the formation of communities and some of the challenges and implications of our work.

5.4.1 The Influence of Time on Communities

Building on the results presented in Section 5.3, we will briefly discuss the communities' evolution over time. Our hypothesis was that, as the behavior of individuals varies from one interval to another, the similarities and/or dissimilarities among them are accentuated. This would have a direct impact on the best way to group such individuals into communities, both in terms of number and composition. Thus, the results presented in Section 5.3 confirm our hypothesis. From them, we could see that the encoded representations resulting from the FL-based and centralized models are grouped in different sets of communities at each interval to better meet the metrics of contact time, similarity and dissimilarity among users.

In all scenarios, we found that to obtain a longer intra-community contact time, the value of K had to be varied periodically, as shown in Table 5.4. By using a higher temporal resolution, i.e., shorter time intervals, we observed a more subtle variation between the K values chosen from one interval to the next. This result, obtained in the NGSIM scenario, reinforces that this setting plays an important role in making the evolution of communities smoother as time progresses. In turn, by using a lower temporal resolution (longer time intervals), the gradual evolution of communities may be lost, and as a result, more drastic changes in the K can be observed. This is because, with this resolution, the samples will aggregate mobility information from a long period in the past, so the impact of recent mobility events is reduced for the detection of communities.

In turn, although the number of users in the scenario is extremely important in determining the average contact time, we confirmed that spatial resolution also has a direct impact on the result of this metric for all pairs of users and, consequently, intra- and inter-community users. Thus, in the NGSIM scenario, whose cells were $40m$x$40m$, we found average contact times of less than $1.4s$ over all the intervals. On the other hand, in the SFC and RT scenarios, whose cells were $300m$x$300m$, we obtained longer contact times. Therefore, both the size of the time interval to update the communities and the size of the region within which users can be in contact have a strong influence on the resulting groupings.

Finally, when we verified that the value of K needs to be varied over time, we consequently proved that a fixed K for all intervals would generate a poor result. This is because both the periodic adjustment of the model and the variation in the behavioral patterns of individuals result in different encoded representations at each interval, which implies new arrangements of communities. So, using one of the probabilistic methods for choosing K exempts us from risking a value for this parameter in each interval. With this, we proved the ability of our proposal to periodically detect communities as time progresses and the behavior of individuals changes.

5.4.2 Challenges and implications

Through our research, we have developed a distributed framework, based on FL, for periodic community detection. Our work uses raw human mobility data to extract spatio-temporal features of displacement and capture the geographical preference of users in urban spaces.

By using FL in our solution, mobility data is kept locally, guaranteeing more privacy for users. Although this data is generated and made easily available on today's mobile devices, and its pre-processing can be carried out locally, the configuration parameters for this stage still need to be determined by a central unit. This is necessary to keep the model's training data (the frequency of generation and the dimensions of the heat maps) consistent across different users.

Since the process of aggregating the FCAE model and detecting communities is dependent on a server, delegating the assignment of these values to it would not incur a significant additional cost. However, it is worth noting that our solution depends on and is highly impacted by the determination of these parameters, as discussed before. In addition, our work does not consider aspects such as city structure and socio-economic factors to detect communities. We purely use raw spatio-temporal mobility data for this task, which in a way simplifies our solution.

In practice, our work can be applied to real-life scenarios in which individuals can benefit from sharing similar behaviors with others, such as spreading messages on opportunistic networks, spreading targeted ads, sharing modes of transport, etc. In addition, the periodic adjustment of communities would keep these services constantly updated, providing even more benefits to their users.

Chapter 6

Case Study: Message delivery in OMSN based on periodic community update

In this chapter, we present a case study based on opportunistic routing using our periodic community detection framework. To this end, we address the concepts related to opportunistic mobile social networks and the characteristics of some of the main opportunistic routing protocols in the literature. Then, we introduce a new protocol, called PCU, based on the Periodic Community Update resulting from the framework proposed in the previous chapter. The objective of this proposed case study is to corroborate the functioning and applicability of our proposal in a real-life scenario.

6.1 Opportunistic Mobile Social Networks

Delay-tolerant networks (DTNs) are characterized by the absence of continuous end-to-end paths between two nodes due to infrastructure unavailability, node mobility, and resource limitations [111]. These characteristics result in network partitioning due to constant topology changes, which leads to long delays in message delivery. Examples of application scenarios for these networks include disaster situations, big events and remote areas where infrastructure is unavailable.

An opportunistic mobile social network (OMSN) is a specific type of DTN. It incorporates aspects of social networks, such as node relationships, into infrastructure-less networks with intermittent connectivity and dynamic topologies. In a real environment, an OMSN can be formed by a set of mobile devices that connect via Bluetooth, such as smartphones, being carried by individuals moving around a city. Since connections are unstable due to mobility, nodes use a mechanism based on storing and

carrying messages until forwarding. Compared to traditional, infrastructure-based networks, OMSNs (and DTNs in general) pose numerous challenges to message delivery, especially regarding delay.

Much effort has been put into improving routing in DTNs over the years [112]. Because many of these networks exhibit social characteristics, their nodes may present long-term relationships and behaviors that are conducive to message delivery [113]. Thus, more recent forwarding solutions, based on OMSNs, have taken advantage of such information, which influences contacts and message-sending opportunities [114]. This social-aware routing enables the selection of more appropriate relays, which improves the data transmission efficiency and reliability, and increases the delivery probability as delay is reduced.

6.1.1 Social metrics for social-aware routing

To estimate the quality of encounters between nodes in OMSNs, some social metrics have been used in recent studies, such as centrality, tie strength, interest, popularity, cumulative contact frequency, community, similarity, fairness, among others [114–116]. Similar to [114], we categorize these metrics into two macro-groups: individual-based social metrics and group-based social metrics. Next, we will briefly present some of the metrics listed in each group.

Centrality, interest and tie strength are among the individual-based social metrics. Basically, centrality measures the importance of a node in the network. This metric can be defined as degree, which measures the number of connections a node has – it also determines the popularity of the node; betweenness, which indicates the ability of a node in a network to be on the shortest path between other nodes; and closeness, which indicates how close a node is to all the others. Interest can be assimilated as an indirect social metric, as it assumes the propensity for friendships and frequent meetings based on the preferences of social nodes. Tie strength quantifies the link between two nodes, such that strong ties are more likely to be activated than weak ties. Its most common indicators are cumulative contact frequency, closeness, longevity, and recency.

As for community, similarity and fairness, they are categorized as group-based social metrics. A community is made up of a set of nodes that are more connected to each other than to outside nodes. Similarity is defined as the amount of common nodes between a pair of nodes. Fairness refers to the equal participation of nodes as traffic load relays, which can impact both the delivery rate and the longevity of network connectivity. For example, in centrality-based routing OMSNs, a small set of nodes (those with the highest centrality) carry the majority of the traffic, which can overload their buffer or even exhaust their energy.

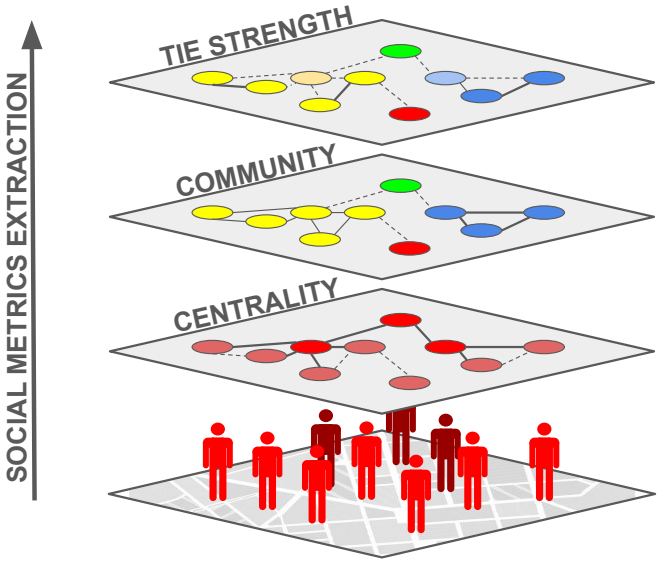


Figure 6.1: Illustration of social metrics extracted from the encounters of individuals moving in urban spaces. In this figure, the node with the highest centrality in the network is identified and highlighted among the others. Similarly, four communities based on encounters and the tie strength between nodes are highlighted. In addition to these, other social metrics can be extracted from the mobility of individuals in an urban setting.

In an urban setting, where individuals carry mobile devices and move using different modes of transportation, social metrics are extracted from their multiple encounters over time. Figure 6.1 illustrates the extraction of social metrics from the encounters of individuals in an OMSN.

6.1.2 Social-aware routing schemes for OMSNs

Based on the above social metrics, several social-aware routing schemes have been proposed [117, 118]. Next, we briefly describe three of them, which have shown a high probability of delivering messages with low delay and cost.

SimBet [119] combines social similarity and betweenness centrality metrics to decide on message forwarding. It explores the underlying social structure of interactions between nodes, using similarity to identify social proximity relationships between nodes and betweenness centrality to identify nodes that can best act as intermediaries in the network. The forwarding decision is based on these two metrics so that the message is sent to nodes that are better positioned in the social network or have greater similarity with the recipient, increasing delivery efficiency in scenarios with intermittent connectivity.

BubbleRap [120] is a well-known routing scheme based on communities and the social centrality of nodes. It explores the idea that nodes tend to form communities

based on the frequency of encounters. The protocol operates on two levels: inter- and intra-community. In the first case, the message is forwarded to nodes with higher global social centrality. In the second, to the nodes with higher local centrality. The combination of centrality and communities allows BubbleRap to be more efficient in terms of delivery and overhead in scenarios with mobility patterns based on social interactions.

EpSoc [121] is a recent protocol that combines the forwarding strategy of the Epidemic protocol with social features such as degree centrality. It adjusts the time-to-live (TTL) of messages based on the centrality of nodes, allowing messages to be forwarded more efficiently to socially active nodes, while a blocking mechanism is implemented to prevent the reception of replicas of messages that have already been discarded. This approach not only increases the message delivery rate but also reduces the overhead and number of forwardings, resulting in superior performance compared to the other protocols, especially in scenarios with larger buffer sizes.

6.2 Opportunistic forwarding based on Periodic Community Update

To corroborate the functioning and applicability of our framework for periodic detection of user communities in a real scenario, we developed a simple social-aware forwarding scheme based on the communities identified over time.

The proposed scheme, called PCU (from Periodic Community Update) is based on the metrics community, tie strength and popularity. The community metric is applied by assigning a label to each user based on data processing performed by our framework. Tie strength is used to compare the propensity of two nodes to find the destination node or its community (i.e., a better forwarder is the one whose tie to the destination user or its community is stronger). The number and variety of meetings a node has, both at the node and community level, in a given time period determines its popularity.

PCU works at two levels of message forwarding: inter- and intra-community. At the inter-community level, we first consider that both the user (or node) carrying the message and the node it finds do not belong to the community of the destination (or target) node. In this case, the message is forwarded primarily based on the tie strength metric, whether the found user has had more encounters with the target user or with other users who belong to the target community. If the tie strength check is negative, the popularity of the found node is checked. In this case, this node must have had more encounters with nodes from other communities (quantity), or have encountered more communities (community-level variety), or, finally, have

encountered more nodes from other communities (node-level variety). Still at the inter-community level, assuming that the user carrying the message finds a node that belongs to the destination community, the message is forwarded without checking metrics, since we assume that users from the same community have a high chance of encountering each other, given their behavioral similarity.

Within the target community, a user carrying the message will only forward it to nodes belonging to the same community. In this case, the encountered node must have a stronger tie to the target user, or be more popular within the community, having encountered a greater variety of nodes or having had more intra-community encounters. Finally, in any chance of sending the message directly to the destination node, the user carrying the message does so without checking any metrics.

To be selected as a message forwarder, the node found must satisfy at least two of the conditions verified for inter-community forwarding and at least one for intra-community forwarding. We injected this restriction to avoid uncontrolled replication of messages in the network. To formalize our socially aware message forwarding scheme for OMSNs, we refer to each of the aforementioned metrics using the following notation:

label(i) refers to the community label of node i;

tie1(i, j) refers to the tie strength between nodes i and j, which is expressed by the number of encounters between them;

tie2(i, c) refers to the tie strength between node i and community c, which is expressed by the number of encounters between the node i and nodes belonging to c;

pop1(i) refers to the inter-community popularity of node i, expressed by the total number of encounters of this node with nodes from other communities;

pop2(i) refers to the inter-community popularity of node i, expressed by the variety of communities found by this node;

pop3(i) refers to the inter-community popularity of node i, expressed by the variety of nodes found belonging to other communities.

pop4(i) refers to the intra-community popularity of node i, expressed by the total number of encounters of this node with nodes in its community;

pop5(i) refers to the intra-community popularity of node i, expressed by the variety of nodes found in its community.

When using the PCU scheme, each node increments these values as encounters occur in the network. Also, whenever communities are updated and a new label is assigned to a user, which is common knowledge to all nodes, these values are reset. Algorithm 3 expresses the working flow of the PCU forwarding scheme.

Algorithm 3: Periodic Community Update (PCU) based forwarding scheme

```
1  for each node found do
2      if label(carrier) != label(target) then
3          if label(found) == label(target) then
4              the message is forwarded to the found node
5          else
6              check = 0
7              check += (tie1(carrier, target) < tie1(found, target))
8              check += (tie2(carrier, label(target)) < tie2(found, label(target)))
9              check += (pop1(carrier) < pop1(found))
10             check += (pop2(carrier) < pop2(found))
11             check += (pop3(carrier) < pop3(found))
12             if check >= 2 then
13                 the message is forwarded to the found node
14     else
15         if label(found) == label(target) then
16             check = 0
17             check += (tie1(carrier, target) < tie1(found, target))
18             check += (pop4(carrier) < pop4(found))
19             check += (pop5(carrier) < pop5(found))
20             if check >= 1 then
21                 the message is forwarded to the found node
```

6.3 Intra-community flooding based on Periodic Community Update

By grouping users into communities, we want to take better advantage of the similar behavioral traits that exist among them. So, inspired by notification mechanisms, but still in OMSN, we designed a scenario in which a message (consider it to be an ad) needs to be delivered to all users in the community within which it was generated.

Different routing schemes in OMSN (and DTNs in general) would be good candidates to accomplish this task. However, our goal in this specific study is not to evaluate decision-making strategies for forwarding and delivering messages. In fact, we are more interested in measuring the quality of communities and how well intra-community users are able to reach each other within a given period of time.

To achieve this, we modified our original PCU routing scheme so that intra-community routing is performed as it occurs in the epideminc. We named this modified scheme IFPCU (Intra-Flooding based on Periodic Community Update). Thus, considering the operation of our periodic community detection framework, in which a new arrangement of communities is formed after each time interval, we define the following scenario:

1. Whenever new community labels are assigned to users, in each community, a random user is chosen to generate a message to every other user in the community.

2. The messages' time to live (TTL) is equal to the size of the interval in which community labels are valid. In other words, when new labels are generated for users, all messages from the previous interval are discarded.

Algorithm 4 details how the IFPCU scheme works.

Algorithm 4: Intra-Flooding based on Periodic Community Update (IFPCU) forwarding scheme

```
1  for each node found do
2      if label(found) == label(target) then
3          the message is forwarded to the found node
4      else
5          check = 0
6          check += (tie1(carrier, target) < tie1(found, target))
7          check += (tie2(carrier, label(target)) < tie2(found, label(target)))
8          check += (pop1(carrier) < pop1(found))
9          check += (pop2(carrier) < pop2(found))
10         check += (pop3(carrier) < pop3(found))
11         if check >= 2 then
12             the message is forwarded to the found node
```

Through IFPCU, messages are transmitted by flooding within communities and only users with a strong social relationship with the target or its community, or who have high popularity in the network are encouraged to participate in the forwarding.

6.4 Experimental Methodology

In this section, we describe the performance evaluation of the proposed schemes, PCU and IFPCU. To this end, we implemented their respective algorithms and evaluated them, using four different scenarios (two real and two synthetic), in the Opportunistic Network Environment simulator, ONE [122].

6.4.1 Experimental Scenarios

To evaluate the proposed schemes, PCU and IFPCU, we performed simulations using two real-world scenarios described in Section 4.1: SFC and RT, for which, we limited the simulation time to 24 hours.

To complement the evaluation, we also used two other synthetic mobility scenarios: Helsinki and Manhattan. The motivation for using these new scenarios is the constancy of mobility records of all users throughout the simulation. That is, throughout the simulation, at well-defined intervals, all users have their location recorded. The lack of this constancy was a problem we faced in the real mobility scenarios mentioned above.

Both the Helsinki and Manhattan scenarios are available for experimentation on ONE and accept the insertion of different modes of transport, including walking. With the exception of trains, the destination of the nodes is defined randomly and their speed and pause time are parameters to be adjusted in advance. Since the real mobility scenarios already represent vehicle mobility, we chose to add only pedestrians in the synthetic scenarios. Thus, we added 200 nodes in the Helsinki scenario and 400 in Manhattan. All nodes were assigned speeds ranging from $0.5m/s$ to $1.5m/s$ and pause times ranging from 0 to $120s$ (both are default pedestrian settings in ONE).

Community detection in synthetic scenarios

To extract communities from the synthetic scenarios, in the same way as we did in the real scenarios (see Chapters 4 and 5), we applied our framework to the mobility synthetic data using the configurations described below.

Regarding the spatial scope, the Helsinki scenario has a total area of $4.5km$x$3.4km$, while Manhattan has $7km$x$7km$. Based on these scopes, we configured the spatial resolution in cells of $150m$x$150m$ for the Helsinki scenario, which resulted in samples (heat maps) of 30 pixels x 24 pixels, and cells of $200m$x$200m$ for Manhattan, resulting in 35 pixel x 35 pixel samples. As for the temporal scope, we limited the simulation

Table 6.1: Scope and spatial and temporal resolution of the Helsinki and Manhattan scenarios

Parameter	Helsinki	Manhattan
Spacial scope	4.5 km x 3.4 km	7 km x 7 km
Cell size	150m x 150m	200m x 200m
Sample shape	32 pixels x 24 pixels	35 pixels x 35 pixels
Temporal scope	4 hours	
Size of the interval t	40 minutes	
Number of intervals	6	

Table 6.2: The FCAE model structure and attributes for the Helsinki and Manhattan scenarios

Parameter	Helsinki	Manhattan
Input shape	32x24x1	35x35x1
Depth	3	
Encoder kernels	128-64-32	
Latent space	100	
Decoder kernels	32-64-128	
Kernels' size	3x3	
Stride	2	
Padding	1	
Activation Function	ReLU	
Loss Function	MSE	

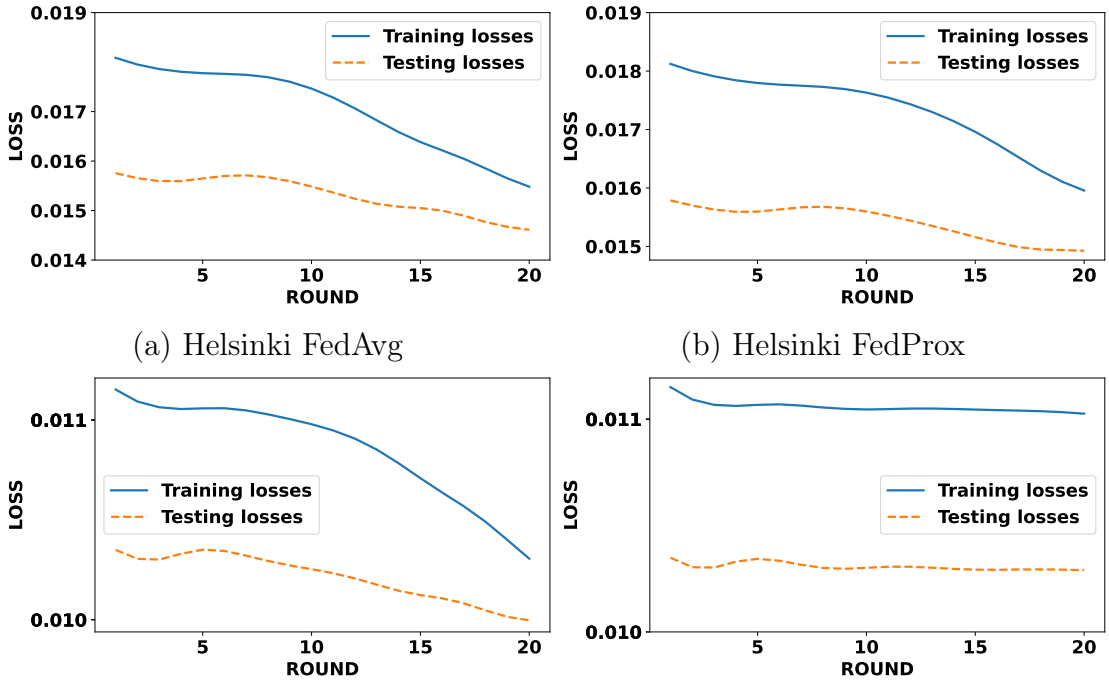
time for these two scenarios to four hours, subdivided into 40-minute intervals for updating the communities. Table 6.1 summarizes the scope and resolution values for these scenarios.

Regarding the structure of the FCAE model, we used the same architecture as the SFC and RT scenarios for the Helsinki and Manhattan scenarios. Although the resolution of the heat maps was not the same, we observed that there was not a significant difference between the shapes of the scenarios. Therefore, we chose to run experiments with a previously tested architecture. The only layer of the architecture that was modified was the input layer, given the shape of the heat maps. Table 6.2 summarizes the FCAE model structure in relation to the sequence of layers and their attributes for these scenarios. Our experiments proved that the architecture was also able to produce satisfactory results for all similarity and dissimilarity metrics in the synthetic mobility scenarios.

As for the hyperparameters, we chose to apply the same values used to train the model for the SFC and RT scenarios, since they produced satisfactory results in terms of convergence and generalization. Table 6.3 summarizes these values, while Figure 6.2 presents the loss curves of the global model trained with the FedAvg and FedProx algorithms for the Helsinki and Manhattan scenarios in window $[0, 1)$. The curves suggest that the resulting models in the Helsinki scenario may present similar

Table 6.3: Hyperparameter settings for the FL-based and centralized approaches for Helsinki and Manhattan

Parameter	FL-based	Centralized
Number of rounds	20	-
Epoch	3	150
Batch size	2	5
Learning rate	0.0005	0.0005
SLI window size (sw)	3	3
Candidates (K)	{1, 2, 3 ..., 20}	



(a) Helsinki FedAvg (b) Helsinki FedProx

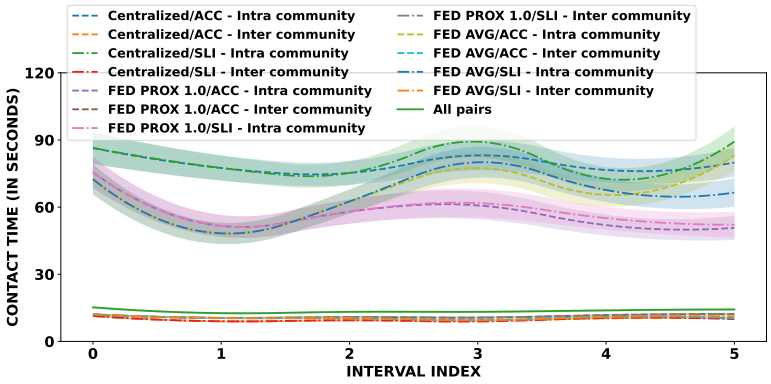
(c) Manhattan FedAvg (d) Manhattan FedProx

Figure 6.2: Loss curves of the global model trained with the FedAvg and FedProx algorithms in window $[0, 1)$ – interval $t0$ – for Helsinki and Manhattan scenarios.

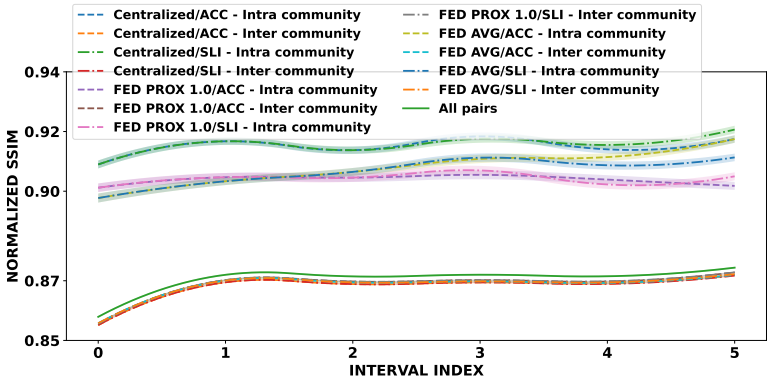
performance and results, given the similarity of their losses. On the other hand, the resulting models in the Manhattan scenario obtained different losses, which indicates a more pronounced difference between their generalization capacity.

Similarly to real mobility scenarios, we analyzed the similarity and dissimilarity metrics of the detected communities. As expected, the similarity metrics between samples and the contact time between users within communities were higher than the average for all users, while the dissimilarity metric was below such average. These results reinforce the ability of our solution to periodically detect good quality communities from different transportation modes. In this case study, we chose to use the results from the SLI strategy in the Helsinki scenario and ACC in Manhattan, given that each obtained greater contact time in the respective scenario. While Table 6.4 presents the *Best* candidate K chosen for each approach over the interval, Figures 6.3 and 6.4 show the contact time and the SSIM, ARI, and MSE metrics for the Helsinki and Manhattan scenarios, respectively.

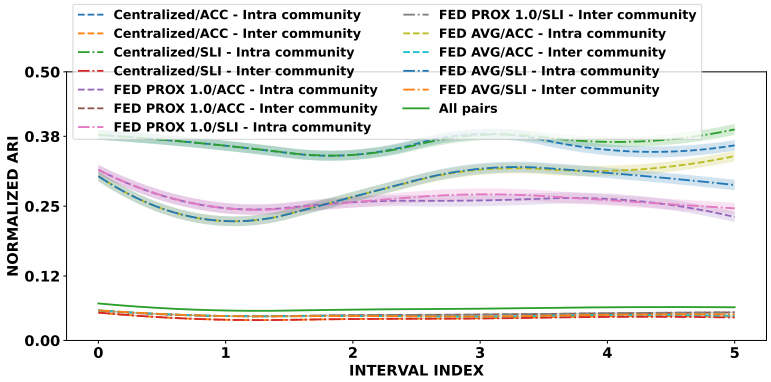
As in real mobility scenarios, we performed FL-based approach experiments with the FedAvg and FedProx aggregation algorithms. In the Helsinki scenario, these algorithms produce very similar results, with alternating gains between the metrics. This result suggests that there is no significant difference between the use of one or the other in such a scenario. On the other hand, in the Manhattan scenario, we observed a more significant gain of the FedProx algorithm compared to FedAvg in



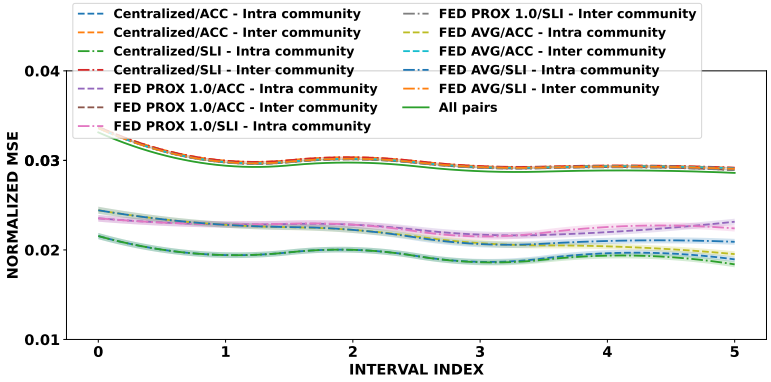
(a) Contact Time



(b) SSIM

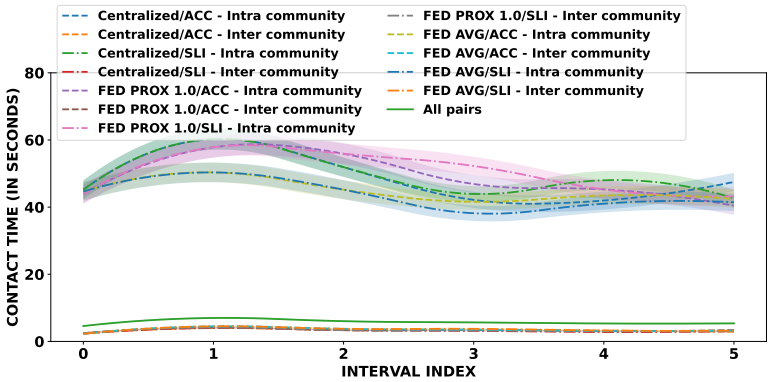


(c) ARI

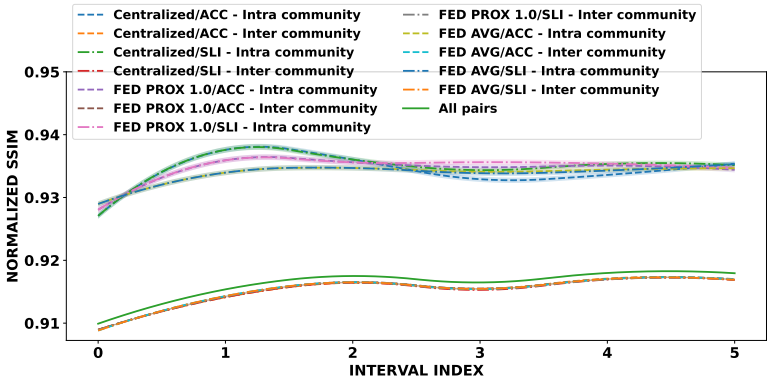


(d) MSE

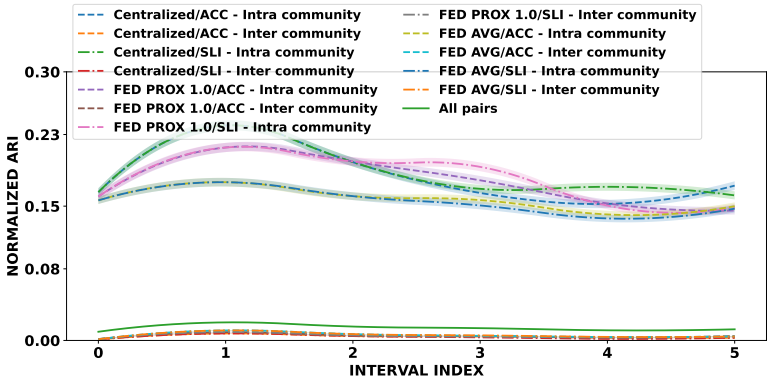
Figure 6.3: Contact time, SSIM, ARI and MSE metrics for the Helsinki scenario using FL-based (FedAvg and FedProx algorithms) and centralized approaches.



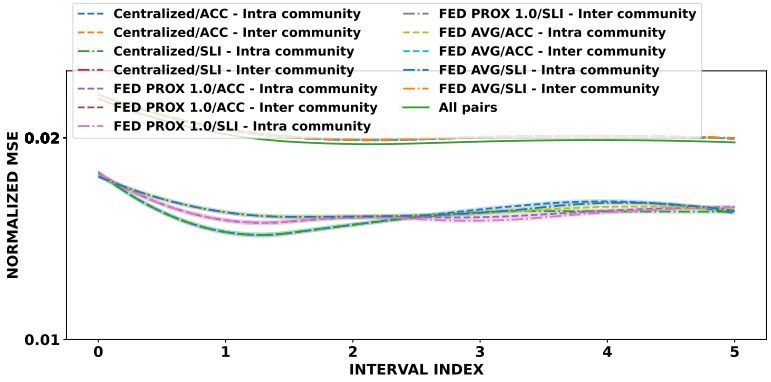
(a) Contact Time



(b) SSIM



(c) ARI



(d) MSE

Figure 6.4: Contact time, SSIM, ARI and MSE metrics for the Manhattan scenario using FL-based (FedAvg and FedProx algorithms) and centralized approaches.

Table 6.4: K chosen for each approach over the intervals in the Helsinki and Manhattan scenarios

K-value chosen (intra- and inter-community)						
	Helsinki			Manhattan		
Interval	FedAvg	FedProx	Centralized	FedAvg	FedProx	Centralized
0	19	20	19	19	20	20
1	20	20	20	18	20	20
2	20	20	20	20	20	20
3	19	19	19	20	20	19
4	19	18	19	20	20	19
5	17	18	18	20	20	20

all metrics. This result indicates the existence of greater heterogeneity between the nodes and data produced in this scenario. Therefore, the model training benefits from the use of an aggregation algorithm that takes these characteristics into account.

Regarding the comparison between all approaches, although there is a clear gain for the centralized approach, once again, the FL-based approaches achieved very similar results, including overlaps. Furthermore, the federated algorithms also achieved satisfactory intra- and inter-community results in relation to the average of each metric for all pairs of users.

6.4.2 Evaluation of Opportunistic forwarding based on Periodic Community Update

To evaluate the performance of the message forwarding scheme in OMSN based on periodic community updates (PCU), we designed simulation experiments on ONE. Below, we describe the experimental setup established for each scenario, the protocols used for comparison, and the evaluation metrics, and then we analyze the results.

Experimental setup

When designing opportunistic communication scenarios in ONE, pairs of source-destination nodes are randomly selected at given time intervals, and new messages can be generated until a certain time or until the end of the simulation.

With this, we chose to ensure that the last message generated on the network had at least its TTL as the time available to be delivered. Thus, in all scenarios, messages are generated up to a time that is equal to the total simulation time minus the TTL of the messages.

As mentioned previously, we limited the simulation time differently for the real and synthetic mobility scenarios. Therefore, we established different configurations based

Table 6.5: Simulation time, inter-message interval and TTL settings for evaluating forwarding schemes in OMSN

Parameter	SFC and RT	Helsinki and Manhattan
Total simulation time (hours)	24h	4h
Inter-message interval	15s to 30s	1s to 3s
1st TTL setting (minutes)	120min (2h)	30min
2nd TTL setting (minutes)	240min (4h)	40min
3rd TTL setting (minutes)	480min (8h)	50min
4th TTL setting (minutes)	720min (12h)	60min

on this parameter. To produce a high message load in the network, we configured the inter-message interval to vary between $1s$ and $3s$ in the scenarios with the shortest simulation time (synthetic) and between $15s$ and $30s$ in the scenarios with the longest duration (real). Furthermore, in each scenario, we evaluated the impact of four different TTL values. Table 6.5 details the simulation times, inter-message interval, and TTL settings for each scenario. We would like to draw attention to the second TTL setting, which is equivalent to the update time of the communities. The PCU result with this setting will be compared to that of the IFPCU scheme, presented in Section 6.4.3.

Finally, the other parameters were configured in the same way for all scenarios. Thus, all nodes transmit messages, whose size varies between 25 KB and 50 KB, at a rate of 1 Mbps within a range of $100m$. In addition, we assigned a 2 GB buffer to each node so that storage capacity would not be an obstacle to message forwarding. Table 6.6 summarizes these parameters.

With these settings, we evaluate the performance of our proposed scheme in comparison with SimBet and BubbleRap, two well-known social context-based protocols (see Section 6.1.2); EpSoc, a recent protocol that injects the notion of social centrality into the functioning of Epidemic; and Epidemic protocols. The Epidemic [123] is inspired by the spread of diseases, in which each node transmits information redundantly and quickly. It is the basis for comparing new protocols in DTN due to its simplicity and effectiveness in guaranteeing delivery in intermittent and mobile networks.

Table 6.6: Common settings for evaluating forwarding schemes in OMSN

Parameter	Value
Messages size	25 KB to 50 KB
Transmission rate	1 Mbps
Transmission range	100 m
Buffer size	2 GB

Evaluation metrics

To measure the performance of our community update-based routing scheme against the above-mentioned mechanisms, we used the four metrics described below:

1. **Delivery probability**: calculated as the success rate of delivered messages. That is, the total number of messages delivered to their respective recipients, divided by the total number of messages created.

2. **Latency**: calculated as the average elapsed time (minutes) from the generation of each message until its delivery to the destination.

3. **Overhead**: defined as the average number of transfers or copies of messages in the network relative to each message created. This metric indicates resource consumption, such as buffer occupancy and energy expenditure for message transmission.

4. **Avg Hops**: counted as the average number of hops for each message delivered.

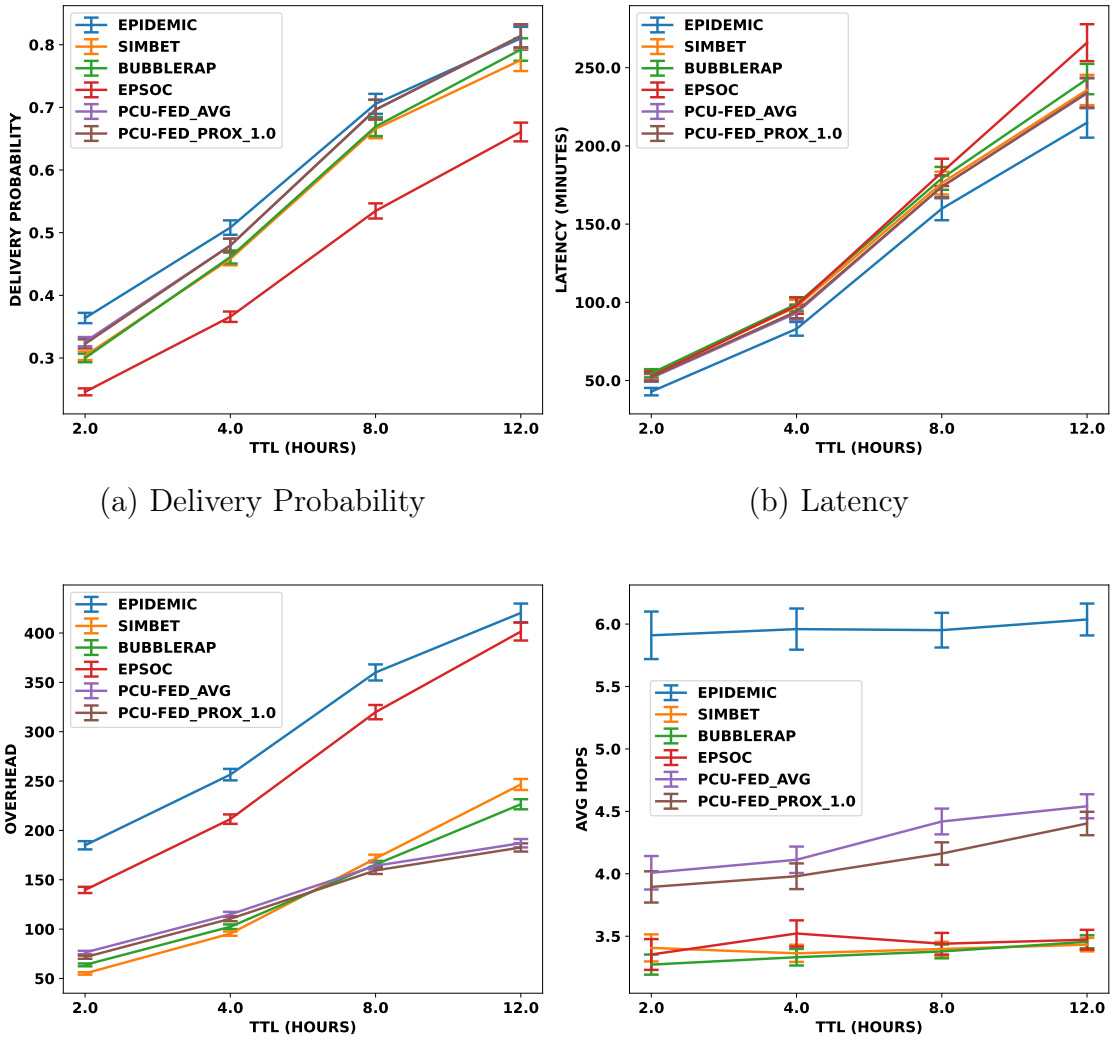
We can say that a routing scheme performs well when it presents a high delivery probability, with low latency, low overhead and a low number of hops required to deliver messages.

Results

The results were obtained for the four scenarios described above with a 95% confidence interval over 10 simulation runs for each TTL. The simulations differ from each other by the random selection of the source-destination pairs of the messages.

Figure 6.5 presents the results of each metric in the SFC scenario. As described above, Epidemic operates naively, without distinguishing between the delivery capabilities of each user, but allowing them to exchange a copy of each stored message. Although expensive, as shown in Figure 6.5(c), which displays the overhead metric, this mechanism allows a high number of messages to be delivered with relatively low latency, as can be seen in Figures 6.5(a) and 6.5(b), respectively.

In turn, the SimBet and BubbleRap protocols, both based on node centrality, have very similar behavior in all metrics of the SFC scenario, as shown in Figure 6.5. Differentiated by the type of centrality – betweenness for SimBet, global and local degree for BubbleRap – and by a second social metric – social similarity and community, respectively – these protocols achieve delivery rates and latency relatively close to the benchmark (Epidemic) with significantly lower overhead and average hops. This result indicates that the use of social metrics in decision-making is beneficial for message forwarding, without a major impact on the delivery rate, when



(a) Delivery Probability

(b) Latency

(c) Overhead

(d) Avg Hops

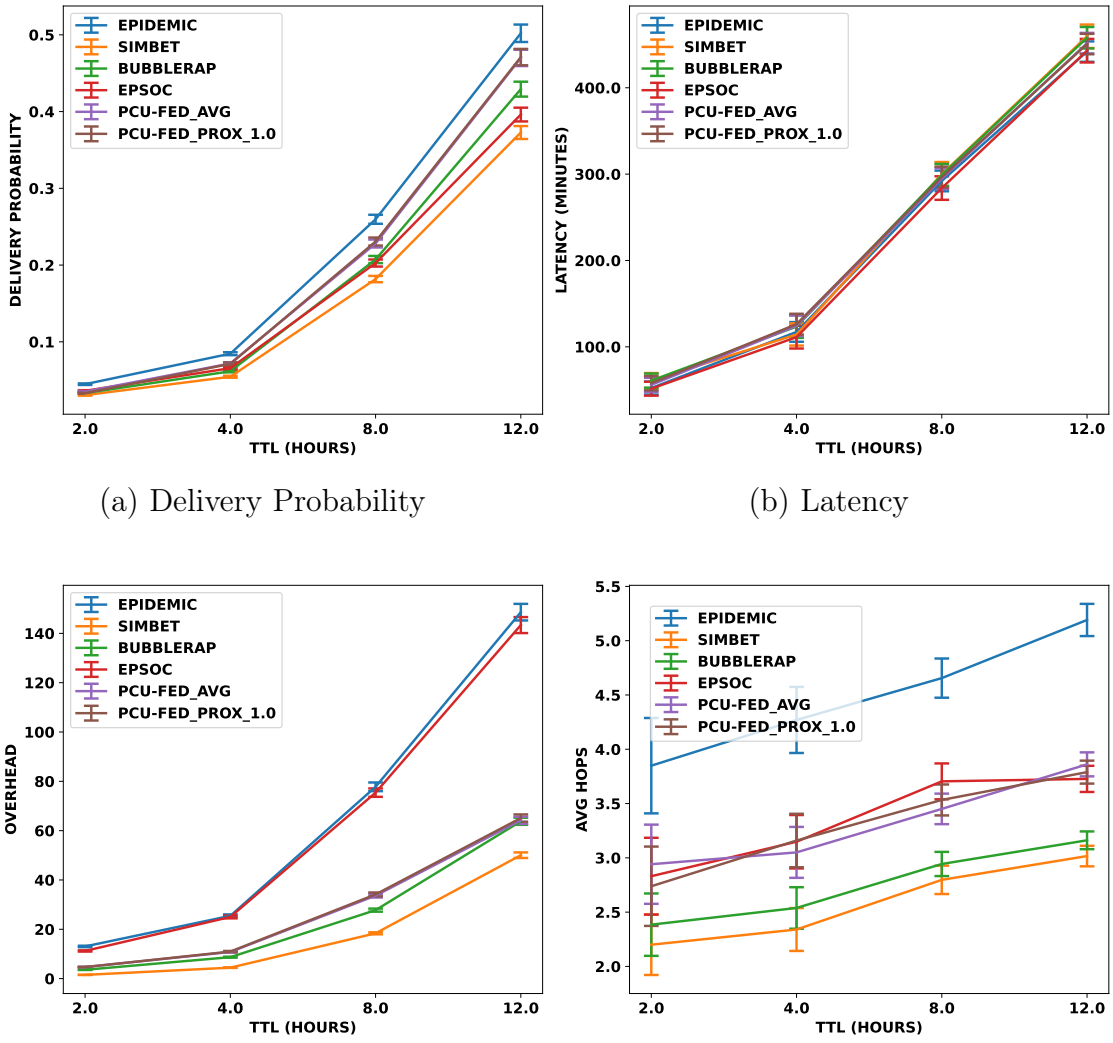
Figure 6.5: Comparison of Epidemic, SimBet, BubbleRap, EpSoc and PCU (FedAvg and FedProx) message forwarding schemes regarding the performance metrics in the SFC scenario.

compared to the benchmark. However, it still does not clarify which of these metrics are the most appropriate for a performance gain, given that both protocols have very similar results.

In contrast, although using a social metric in its operation, EpSoc presents the lowest delivery rate in the SFC scenario, with a latency very close to that of the other protocols. By modifying the TTL of messages that are received by nodes with greater centrality, but allowing forwarding to all nodes when they come into contact, it was expected that the protocol would combine the advantages of both the benchmark and the use of social metrics in decision-making. However, we observe that this protocol maintains only the disadvantages of Epidemic in the SFC scenario, by obtaining such a high overhead (Figure 6.5(c)). The low performance of EpSoc is

explained because, according to the authors [121], for messages with short TTL (less than 1.5 days), exploring social features will not be very advantageous, especially to further reduce such TTL, due to the rapid discarding of messages. Given that the scenarios proposed in our study involve messages with a maximum TTL of 4 hours, we did not anticipate that EpSoc would outperform the other protocols.

The low influence of the social centrality metric on EpSoc when the TTL is low can be seen in the protocol's resulting latency. Considering that in its operation, nodes with higher centrality reduce the TTL of messages, a lower delivery latency would be expected if these nodes delivered most messages. However, we see a higher latency than the benchmark. This indicates that most of the delivered messages take alternative and slower routes.



(a) Delivery Probability

(b) Latency

(c) Overhead

(d) Avg Hops

Figure 6.6: Comparison of Epidemic, SimBet, BubbleRap, EpSoc and PCU (FedAvg and FedProx) message forwarding schemes regarding the performance metrics in the RT scenario.

When designing the case study, we hypothesized that a routing scheme based on the communities detected by our framework would be impacted mainly in terms of latency. This hypothesis was motivated mainly by the objective of the framework, which, by identifying communities, would bring together users with greater similarity and, therefore, social interactions, which could facilitate message delivery. However, it is necessary to consider that the proposed experiment produces messages that must be delivered between different communities. Therefore, it is expected that this metric will be affected since interactions between users from different communities tend to be lower.

Thus, as shown in Figure 6.5(b), referring to latency in the SFC scenario, our PCU scheme, both resulting from FedAvg and FedProx, ties with the other social context protocols in the confidence interval, being surpassed only by Epidemic. Regarding the overhead generated by the PCU, we observed that it is very close to that of SimBet and BubbleRap, even when it has a worse result (TTL = 2h and TTL = 4h). In comparison with these protocols, it was expected that the PCU would have a higher overhead since it considers more candidates as eligible for message forwarding, which is also expressed in the average number of hops (Figure 6.5(d)). Although this is confirmed, there is a trade-off in delivery rate, given that PCU outperforms other social context protocols, achieving a result even closer to the benchmark in the SFC scenario (Figure 6.5(a)).

Figure 6.6 presents the results of each metric in the RT scenario. Among the four scenarios presented in our study, we consider this to be the most challenging for delivering messages via DTN. Therefore, the delivery rate considered low for all routing schemes (below 50%) is justified by two reasons. First, the low number of users at all intervals in a relatively large spatial area limits their contact opportunities. The second is the variability of this number, both from one interval to another and within each interval. As explained previously, this inconsistency occurs due to the lack of mobility records for certain nodes over certain periods of time.

Despite this, we observe in Figure 6.6(a) that PCU schemes obtain a higher delivery rate than BubbleRap, which is also based on community and social context. Furthermore, our scheme also outperforms the SimBet and EpSoc protocols on this metric. In comparison with Epidemic, considered the upper limit in this metric, all the others performed below, as expected, especially in a scenario with such limited contact opportunities.

However, when we check the other metrics, especially the overhead (Figure 6.6(c)), we notice that to achieve this gain, Epidemic injects many more copies of messages into the network than the other schemes, except for EpSoc, which, in this metric, is practically equal to Epidemic. In this regard, we observed that EpSoc's high overhead allowed it to surpass SimBet's delivery rate in the RT scenario, as shown in Figure

6.6(a). That is, we attribute this gain of EpSoc to its forwarding naivety rather than to its use of the centrality metric, since, again, the latency of this protocol is similar to that of the other schemes.

Regarding the protocols based on social context, including our PCU scheme, we find that the results are in agreement with those obtained in the previous scenario. We highlight the overhead resulting from PCU, which, again, is slightly higher than that of the other social context protocols, but considerably lower than that of Epidemic. This result shows the balance of our proposal regarding message forwarding flexibility, which guarantees a considerably low overhead with a high delivery rate.

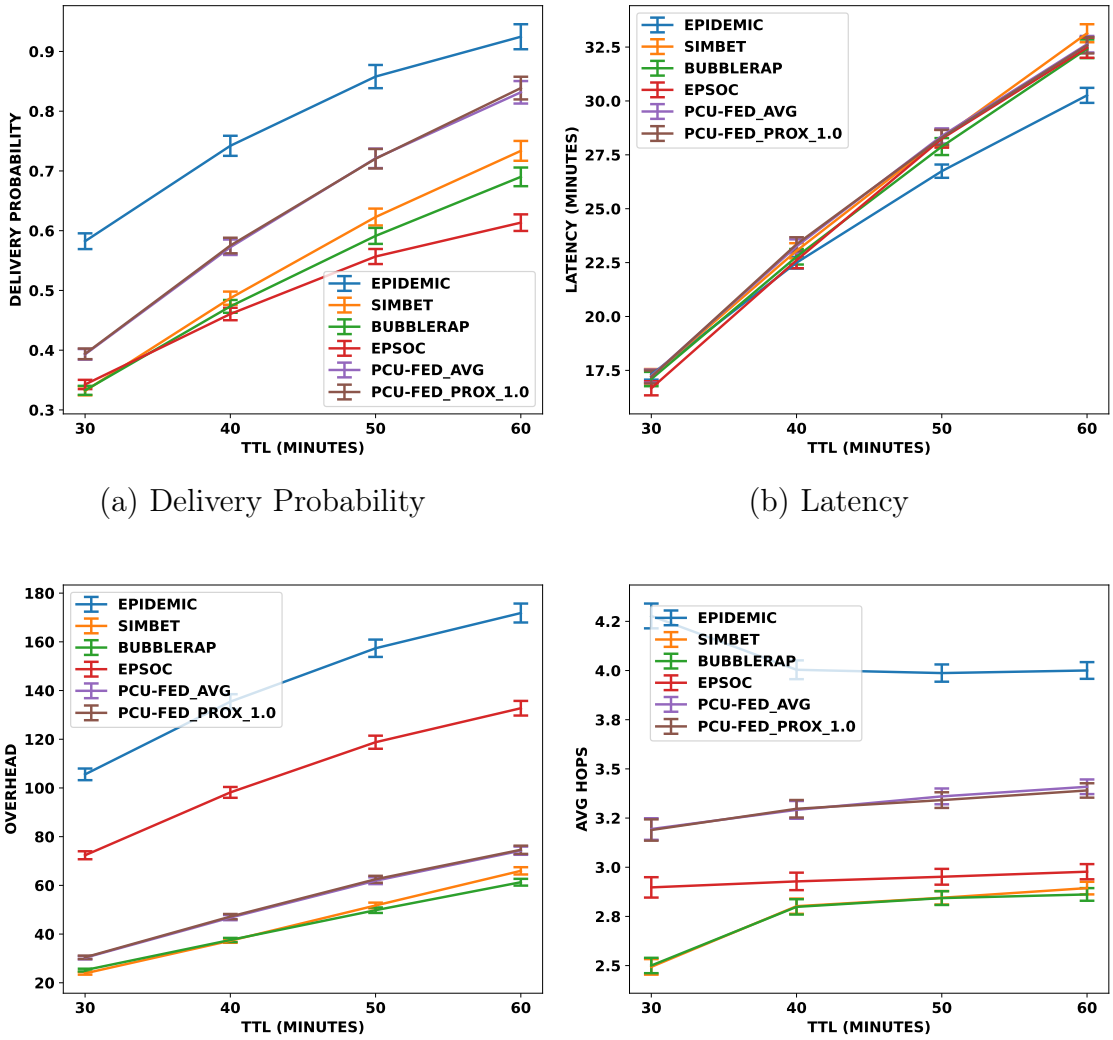
Just as importantly, these metrics also show the quality of the communities detected by our framework in both real-world mobility scenarios. This is because PCU works based on ties between users, between users and communities, and the popularity of users with other communities or within their own. Therefore, the high delivery rate (closest to the benchmark) with low overhead and latency also similar to the benchmark implies that the ties and popularity of users and communities were really high, confirming the quality of the division into communities.

We also observed that the quality of the communities obtained by each aggregation algorithm (FedAvg and FedProx) does not vary so much as to significantly impact the PCU performance. On the contrary, the values obtained in each metric are quite similar when comparing the two algorithms.

Moving on to synthetic mobility scenarios, considered more stable, as there is no variation in the number of users on the network, Figures 6.7 and 6.8 present the results of the metrics for the Helsinki and Manhattan scenarios, respectively. As observed in the results of real scenarios, Epidemic tends to achieve a higher delivery rate than other schemes, which occurs at the cost of greater consumption of resources, such as storage capacity and energy. This is clear in the ratio between the number of copies (retransmissions) per message delivered on the network.

In the Helsinki scenario, we observe in Figure 6.7(a) that again the social context protocols outperformed EpSoc in the delivery rate. We also observed that this protocol has a high overhead (Figure 6.7(c)) with latency tied to that of all protocols (Figure 6.7(b)), suggesting, once again, that the social centrality metric is having minimal influence on message delivery.

Regarding PCU performance, Figure 6.7(c) shows that our scheme achieved a slightly higher overhead than SimBet and BubbleRap, but with a much more significant gain in delivery rate compared to these protocols (Figure 6.7(a)). Furthermore, in the average number of hops, although PCU obtained the second-highest average, the difference to the lowest value is less than one hop and therefore has negligible impact on the overall picture. These results reinforce that the communities resulting from our framework combined with the choice of metrics for forwarding decision-



(a) Delivery Probability

(b) Latency

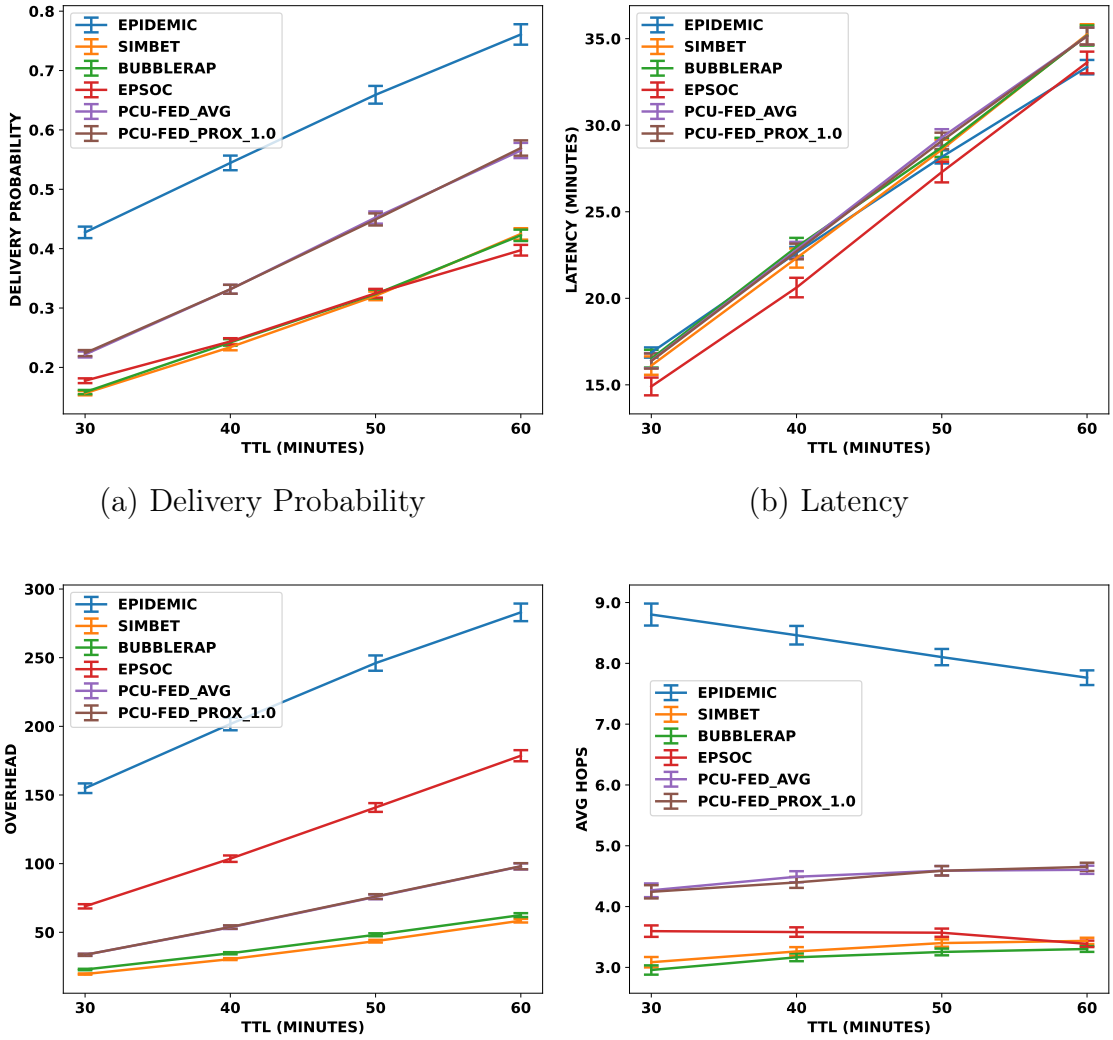
(c) Overhead

(d) Avg Hops

Figure 6.7: Comparison of Epidemic, SimBet, BubbleRap, EpSoc and PCU (FedAvg and FedProx) message forwarding schemes regarding the performance metrics in the Helsinki scenario.

making in our scheme result in higher efficiency than these protocols. Especially when compared to BubbleRap, which also performs community calculations.

Although BubbleRap updates communities periodically, it does not clearly define the update interval. In addition, the protocol uses the concept of k-clique (a complete graph of size k), in which a community is defined as a set of adjacent k-cliques. In this context, two k-cliques are adjacent if they share k-1 nodes. The problem with this approach lies in the rigid definition of community; that is, for a group of nodes to be considered a community, they must meet the criterion of containing at least one k-clique, with k fixed. This excludes nodes that are "close" in terms of social interaction or connectivity but that do not exactly satisfy this rigid structure. To circumvent this problem, the authors make the pragmatic decision to allow the



(a) Delivery Probability

(b) Latency

(c) Overhead

(d) Avg Hops

Figure 6.8: Comparison of Epidemic, SimBet, BubbleRap, EpSoc and PCU (FedAvg and FedProx) message forwarding schemes regarding the performance metrics in the Manhattan scenario.

existence of single-node communities so that each node belongs to at least one community. This approach, however, does not allow for a middle ground between communities of size 1 and k, which may disregard the social proximity of any group of $\bar{k}$ nodes, where $1 < \bar{k} < k$.

Finally, in the Manhattan scenario, we found that all metrics behaved very similarly to the previous scenario. The exception was the EpSoc protocol, which achieved a delivery rate very close to the SimBet and BubbleRap protocols, as shown in Figure 6.8(a), and no longer tied with the others in latency but had the lowest one among all (Figure 6.8(b)). However, given the insignificant difference in latency compared to the others, this gain is still not a clear indication that the protocol benefited from the use of the social centrality metric in this scenario.

The better performance of PCU compared to SimBet and BubbleRap is evident by its higher delivery rate, as shown in Figure 6.8(a), at the cost of overhead and an average number of hops that, although higher, is very close to those of these protocols (Figure 6.8(c) and Figure 6.8(d), respectively). This once again highlights the good choice of social metrics for decision-making in PCU and the quality of the communities resulting from our framework.

6.4.3 Evaluation of intra-community flooding based on Periodic Community Update

To evaluate the performance of the intra-community flooding scheme in OMSN based on periodic community updates (IFPCU), we designed simulation experiments on ONE. Below, we describe the experimental setup established for each scenario, the protocols used for comparison, and the evaluation metrics, and then we analyze the results.

Experimental setup

As described in Section 6.3, for the intra-community message flooding experiments, we use the interval in which communities are valid as both the inter-message interval and the TTL. Furthermore, within each new community, a user is randomly chosen to generate a message for each user in its community. Table 6.7 summarizes the simulation times, message intervals, and TTL for all proposed scenarios. The remaining network parameters were configured in the same way as in the previous section, therefore, Table 6.6 also applies to the intra-community flooding experiments.

Table 6.7: Simulation time, inter-message interval and TTL settings for evaluating intra-community floading schemes in OMSN

Parameter	SFC and RT	Helsinki and Manhattan
Total simulation time (hours)	24h	4h
Inter-message interval	4h	40min
TTL setting (minutes)	240min (4h)	40min

With these settings, we evaluated the performance of our proposed scheme in comparison with the Epidemic protocol, using the same message generation mechanism. For each protocol, the communities resulting from the FedAvg and FedProx aggregation algorithms were experimented. We chose not to compare it with other protocols, especially BubbleRap because its definition of communities is different. Therefore, it would be an unfair comparison since the origin and destination of the messages are defined according to the communities resulting from our framework.

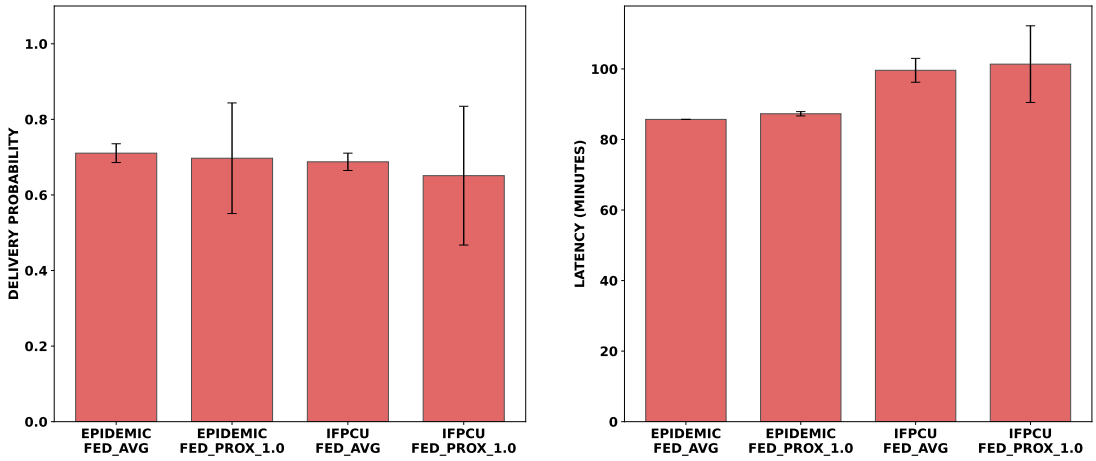
Evaluation metrics

The evaluation metrics are the same as those described in Section 6.4.2.

Results

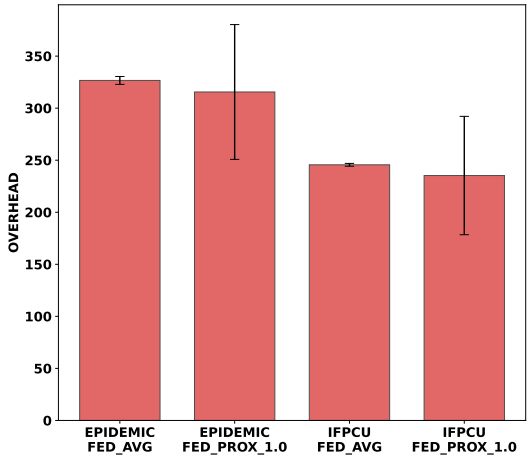
The results were obtained for the four scenarios described above with a 95% confidence interval over 10 simulation runs for each protocol. The simulations differ from each other by the random selection of the source node within each community.

Figures 6.9 and 6.10 show the results of each metric for the SFC and RT scenario, respectively. From these results, we observe that, in terms of latency (Figures 6.9(b) and 6.10(b)), there was no significant difference compared to the previous experiment for both Epidemic and IFPCU (compared to PCU) forwarding schemes. In both scenarios, this comparison is made with the 2nd TTL of Table 6.5. In the previous

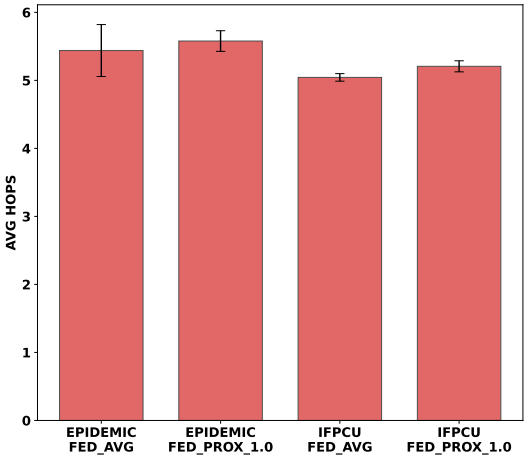


(a) Delivery Probability

(b) Latency

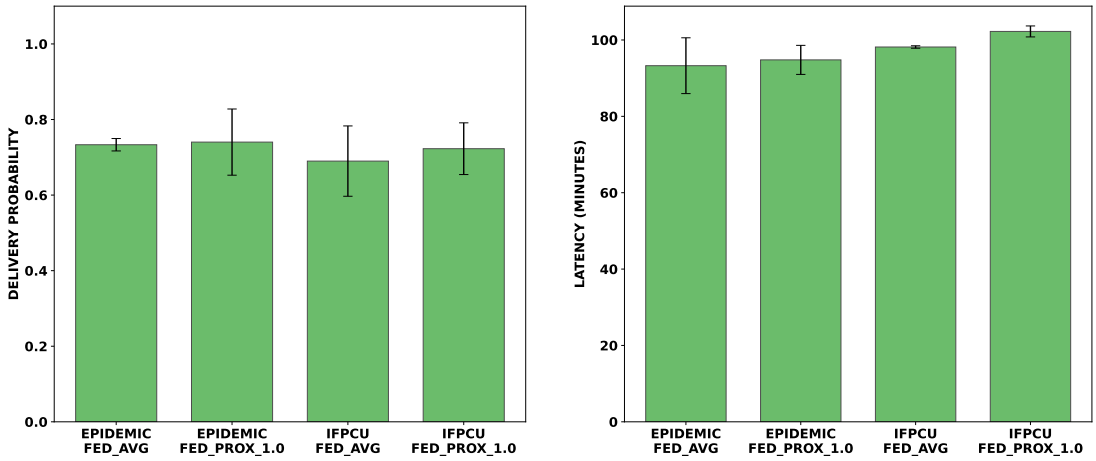


(c) Overhead



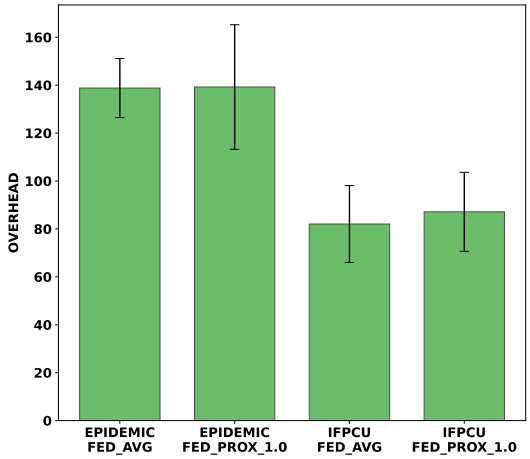
(d) Avg Hops

Figure 6.9: Comparison of the Epidemic and IFPCU schemes, both using communities resulting from the FedAvg and FedProx algorithms, regarding the performance metrics in the SFC scenario.

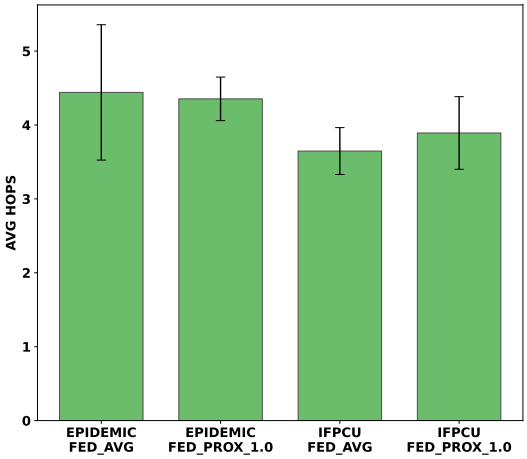


(a) Delivery Probability

(b) Latency



(c) Overhead



(d) Avg Hops

Figure 6.10: Comparison of the Epidemic and IFPCU schemes, both using communities resulting from the FedAvg and FedProx algorithms, regarding the performance metrics in the RT scenario.

experiment, the source-destination pairs of the messages included users from different communities. In the current experiment, even though the messages are destined for users in the same community, both protocols allow them to travel through users in different communities, which explains the similarity in latency.

Although the IFPCU scheme differs from the PCU in intra-community decision-making, where the former is more flexible than the latter, which facilitates message delivery in this context, this result shows a certain stability in the latency. Such stability could suggest a lower performance compared to the previous experiment, given that messages are now generated within communities. However, through Figures 6.9(a) and 6.10(a), we observe an increase of at least 17% (IFPCU with FedProx in the SFC scenario) and 61% (IFPCU with FedAvg in the RT scenario) in the average delivery rate compared to the previous experiment, not only for our mechanism, but also for the Epidemic. Although these are different message

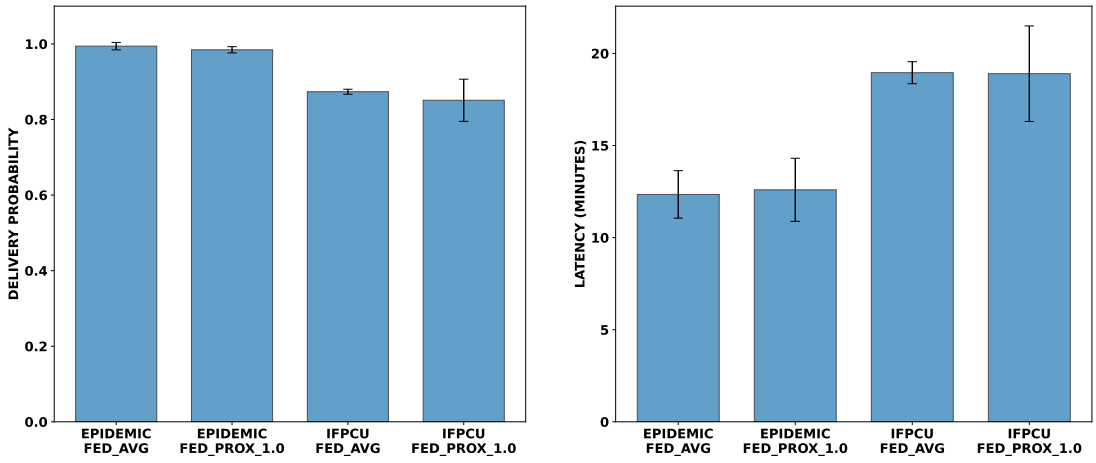
generation patterns, this gain indicates that both benefit from the fact that messages are generated "closer" to the destinations, which increases their probability of delivery.

Regarding the overhead generated by both Epidemic and IFPCU in both scenarios, we observe in Figures 6.9(c) and 6.10(c) a considerable increase in comparison with the previous experiment. This increase is explained by the difference in the message generation pattern. That is, with more messages available in the buffer, the number of transmissions at each contact opportunity increases. Therefore, since in the previous experiment only one new message was generated at a time, network overhead tended to be lower. In particular, the higher overhead of IFPCU compared to PCU is explained by two combined factors. First, the flooding of messages within the target community. Second, the flexibility for some users outside the target community to participate in the forwarding. Despite this increase when compared to PCU, IFPCU still performs better than the Epidemic in this metric. In fact, we proposed conducting this experiment with the Epidemic both to evaluate its performance on communities and to have a solid basis for comparison.

Finally, regarding the number of hops, we observe in Figures 6.9(d) and 6.10(d) that Epidemic remained very close to the result of the previous experiment, while the average of IFPCU increased by around one hop in relation to PCU. From this, we conclude that this increase occurs due to the part of the protocol that works through flooding. That is, as the chances of delivery increase due to this flexibility, more nodes tend to receive a copy of the message, impacting the average number of hops.

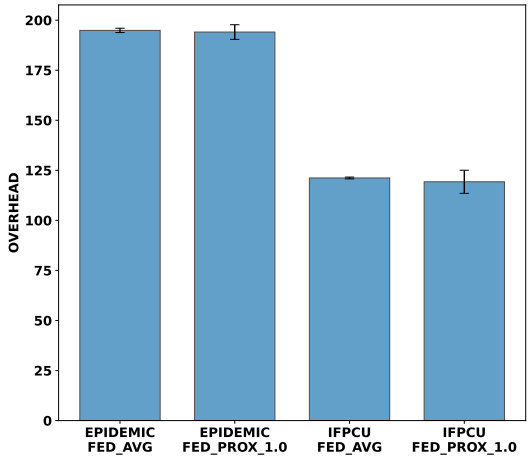
In turn, Figures 6.11 and 6.12 present the results of each metric for the synthetic scenarios, Helsinki and Manhattan, respectively. Unlike real mobility scenarios, we observe in Figures 6.11(b) and 6.12(b) a decrease in delay for both mechanisms compared to the previous experiment. This, in advance, corroborates our hypothesis that messages destined to users in the same community are delivered faster. To reinforce this hypothesis, both in the Helsinki and Manhattan scenarios, there were increases in the average delivery rate, compared to the previous experiment, of at least 27% and 41%, respectively, as shown in Figures 6.11(a) and 6.12(a), both at IFPCU with FedProx. Although all scenarios, both real and synthetic, improved the delivery rate, we consider the extensive increase in this metric and the reduction in latency in the synthetic scenarios to the stability of the nodes in the network. As mentioned previously, the lack of mobility records and consequent inactivity of some nodes in real scenarios for long periods significantly impacts the functioning of the network, further reducing its connectivity.

Regarding the overhead, shown in Figures 6.11(c) and 6.12(c), we reiterate the analysis discussed in real mobility scenarios. That is, in comparison with the previous experiment, for both mechanisms the increase occurs mainly due to the message generation pattern. Thus, at the beginning of an interval, instead of just one, the

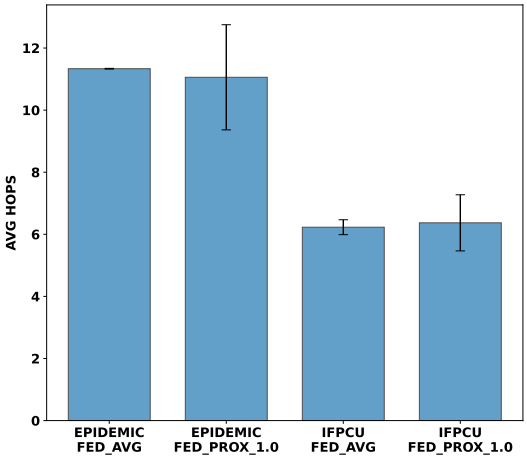


(a) Delivery Probability

(b) Latency



(c) Overhead

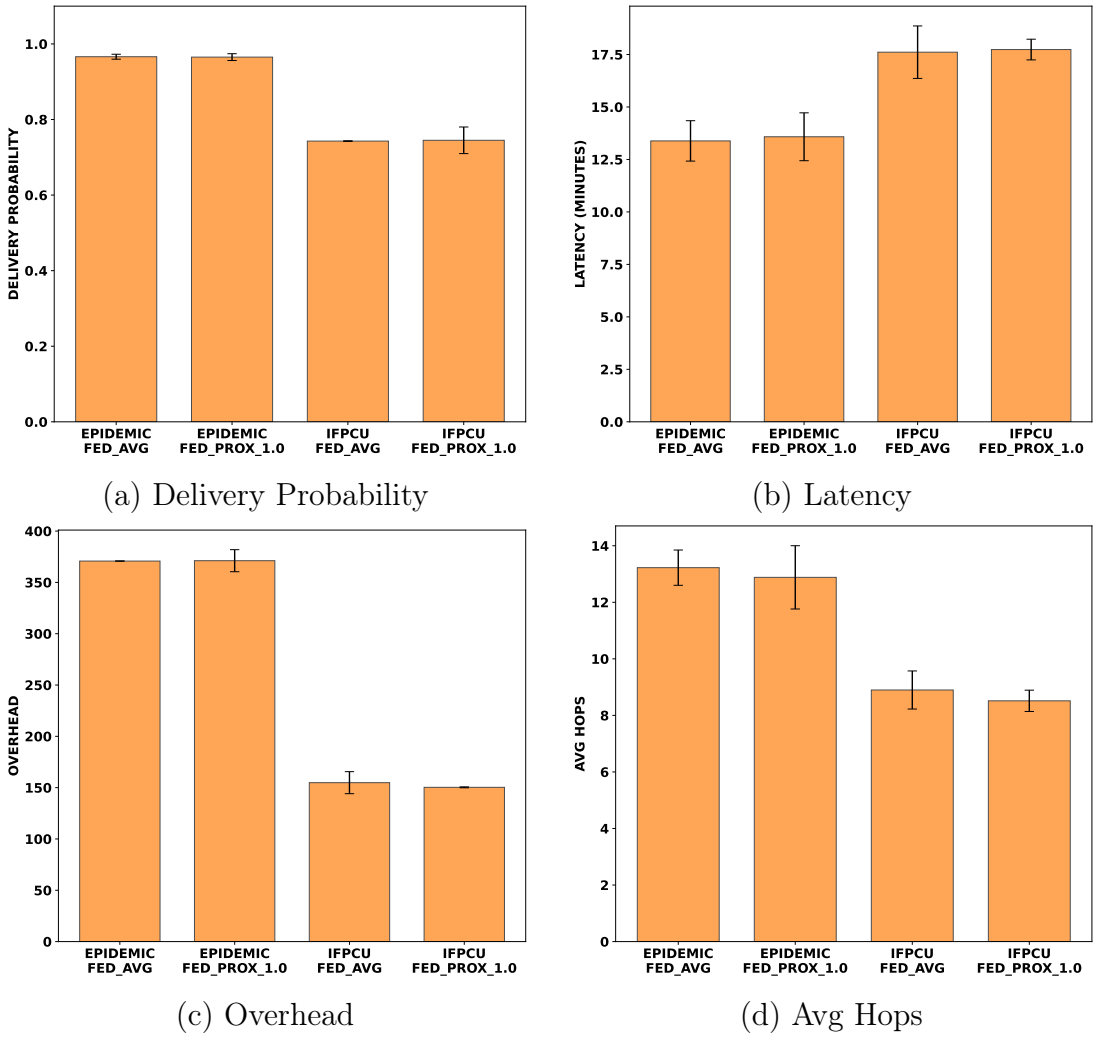


(d) Avg Hops

Figure 6.11: Comparison of the Epidemic and IFPCU schemes, both using communities resulting from the FedAvg and FedProx algorithms, regarding the performance metrics in the Helsinki scenario.

selected node generates several messages. In its next contact with another node, it has the opportunity to send many of them, depending on the duration of the contact. In contrast, in the previous experiment, a time interval elapses until another node creates a single message.

On the one hand, this difference in the message generation pattern justifies the different behavior of the overhead metric between the two experiments. Therefore, it would be reasonable to state that the increase in the delivery rate also occurs due to this pattern. However, although messages are created in batches and their quantity is smaller in the current experiment, in both cases all messages have the same chance of being delivered, given that none are discarded before the end of their TTL. Furthermore, although we are comparing the performance of the metrics in both experiments, our goal is not to consider them equivalent. Rather, we want to emphasize how these metrics behave in each situation, highlighting the use of



(a) Delivery Probability (b) Latency

(c) Overhead (d) Avg Hops

Figure 6.12: Comparison of the Epidemic and IFPCU schemes, both using communities resulting from the FedAvg and FedProx algorithms, regarding the performance metrics in the Manhattan scenario.

community information resulting from our framework.

As for the average number of hops (Figures 6.11(d) and 6.12(d)), although we observe an increase in relation to the previous experiment, this number of hops is similar to the average number of users per community in each scenario (ratio between the total number of users and the number of communities in Table 6.4). This indicates that messages from/to a community, although they may travel other routes, transit internally between several nodes before being delivered. Even if this occurs, which increases the number of hops, if we combine this result with the average delay, we see that these contacts occur more frequently than the inter-community contacts in the previous experiment.

Thus, both real and synthetic mobility scenarios have shown to be strongly benefited by employing behavioral similarity information between users for message delivery. Although the overhead has increased in relation to the previous experiment

for both mechanisms, we found that the benchmark (Epidemic protocol), whose decision-making was not modified, also presented a significant improvement in the delivery rate and latency. This result confirms the quality of the communities detected by our framework.

Through the case study presented in this chapter, we verified the ability of our framework to identify groups of users that actually present behavioral similarities, and we took advantage of this ability to simulate a usage application. Our results prove that the detected communities assist in the process of delivering intra- and inter-community messages in OMSN, generating better results than well-known mechanisms in the literature based on social metrics.

Chapter 7

Conclusion and Future Works

In this work, we proposed a methodology for the detection and periodic grouping (in communities) of individuals with behavioral similarities. For this, we built a Full Convolutional Autoencoder deep learning model, based on [22] proposal for mobility feature extraction, and train it using Federated Learning. By using FL, we focused on data privacy preservation as well as taking advantage of features like collaborative learning, minimal update communication, immediate local training (as soon as the data is released), minimal retention of information, and non-centralization of data by the server [26].

From each user's mobility samples, the proposed model generates encoded representations, which are used as input to the Model-Based Clustering (MBC) algorithm for grouping these individuals into communities. In our proposal, we verified the performance of the probabilistic methods AIC and BIC for dinamically defining the number of communities. We also experimented with different sizes for the training dataset and analyzed the evolution of communities over time. Finally, we compared the FL-based training approach to the centralized one for the community identification process. As evaluation metrics, we analyzed the average contact time and three other metrics of spatial similarity and dissimilarity among intra- and inter-community users.

To evaluate our proposal, we conducted experiments with three sets of real mobility data. When analyzing the results, we found that the AIC method performed better in comparison with the BIC in choosing the number of communities, as it produces better results in the evaluation metrics in most cases. Regarding the strategies that define the size of the training sample set, we found that both ACC and SLI produced models whose results met the proposed evaluation metrics. While the ACC strategy generated better results in scenarios with greater spatial extent and lower temporal resolution, the SLI strategy was more efficient in scenarios with the opposite characteristics. Furthermore, we found that the FedAvg and FedProx federated aggregation algorithms benefit from the use of each of these strategies.

While the former takes advantage of the greater quantity and diversity of ACC samples, the latter benefits from the reduction of local data heterogeneity resulting from the SLI strategy. These factors favor the convergence of the respective models. We found that, in fact, the composition of communities varies from one interval to another, confirming that the individuals' behavioral patterns change over time and, consequently, their similarities and dissimilarities. Finally, when comparing the training approaches, we observed very close values in the evaluated metrics results and a high percentage of overlapping both in the means and in the confidence intervals. Our results therefore indicate a high ability to identify communities from our proposal without major losses compared to a traditional training approach, reinforcing the advantages of using Federated Learning.

We also conducted a case study in which we leveraged the framework and the detected communities to perform message delivery in Opportunistic Mobile Social Networks (OMSN). To this end, we developed a simple message forwarding mechanism based on community, tie strength between users and communities, and intra- and inter-community user popularity. We evaluated the mechanism in two real mobility scenarios and two synthetic mobility scenarios and compared its performance to that of classic protocols in DTN networks. We found that when community information is used, evaluation metrics, especially delivery rate, latency and overhead, are significantly improved.

In future work, inspired by the results of one of the real mobility scenarios, we intend to further study the scarcity of users and mobility samples to train the model in each interval. This implies establishing a threshold below which these quantities can be considered insufficient. With this, we want to improve our periodic community detection framework, making it functional for urban space services.

We also want to combine the proposed framework with services that can directly benefit from identifying groups with similar behavioral patterns, such as Intelligent Transportation Systems (ITS) [124]. Our goal is to evaluate metrics that demonstrate this benefit, such as travel time, delay time, vehicle density and speed, and traffic volume [125].

Finally, we seek to expand the applicability of the proposed framework by evaluating its potential in other domains involving the analysis of collective behavior, such as urban planning and emergency management, for example. In doing so, we hope to contribute to the development of smarter and more adaptable solutions for connected cities.

References

[1] CAPDEVILA, I., ZARLENGA, M. I. "Smart city or smart citizens? The Barcelona case", *Journal of strategy and management*, v. 8, n. 3, pp. 266–282, 2015.

[2] ZUBIZARRETA, I., SERAVALLI, A., ARRIZABALAGA, S. "Smart city concept: What it is and what it should be", *Journal of Urban Planning and Development*, v. 142, n. 1, pp. 04015005, 2016.

[3] YU, Q., LI, W., YANG, D., et al. "Mobile phone data in urban commuting: a network community detection-based framework to unveil the spatial structure of commuting demand", *Journal of Advanced Transportation*, v. 2020, pp. 1–15, 2020.

[4] CHEN, J., XIAO, Z., WANG, D., et al. "Stay of Interest: A dynamic spatiotemporal stay behavior perception method for private car users". In: *2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pp. 1526–1532, New York, 2019. IEEE.

[5] WANG, X., LIU, Y., ZHOU, X., et al. "A point-of-interest recommendation method exploiting sequential, category and geographical influence", *ISPRS International Journal of Geo-Information*, v. 11, n. 2, pp. 80, 2022.

[6] WANG, A., ZHANG, A., CHAN, E. H., et al. "A review of human mobility research based on big data and its implication for smart city development", *ISPRS International Journal of Geo-Information*, v. 10, n. 1, pp. 13, 2020.

[7] FERRETTI, M., BARLACCHI, G., PAPPALARDO, L., et al. "Weak nodes detection in urban transport systems: Planning for resilience in Singapore". In: *2018 IEEE 5th international conference on data science and advanced analytics (DSAA)*, pp. 472–480, New York, 2018. IEEE.

[8] KRAEMER, M. U., YANG, C.-H., GUTIERREZ, B., et al. "The effect of human mobility and control measures on the COVID-19 epidemic in China", *Science*, v. 368, n. 6490, pp. 493–497, 2020.

[9] PAPPALARDO, L., VANHOOF, M., GABRIELLI, L., et al. "An analytical framework to nowcast well-being using mobile phone data", *International Journal of Data Science and Analytics*, v. 2, n. 1, pp. 75–92, 2016.

[10] MITTAL, R., BHATIA, M. "Classification and comparative evaluation of community detection algorithms", *Archives of Computational Methods in Engineering*, v. 28, pp. 1417–1428, 2021.

[11] JIN, D., YU, Z., JIAO, P., et al. "A survey of community detection approaches: From statistical modeling to deep learning", *IEEE Transactions on Knowledge and Data Engineering*, v. 35, n. 2, pp. 1149–1170, 2021.

[12] LIND, A., HADACHI, A., PIKSARV, P., et al. "Spatio-temporal mobility analysis for community detection in the mobile networks using CDR data". In: *2017 9th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, pp. 250–255, New York, 2017. IEEE.

[13] ALESSANDRETTI, L. "What human mobility data tell us about COVID-19 spread", *Nature Reviews Physics*, v. 4, n. 1, pp. 12–13, 2022.

[14] TAJEUNA, E. G., BOUGUESSA, M., WANG, S. "Modeling and predicting community structure changes in time-evolving social networks", *IEEE Transactions on Knowledge and Data Engineering*, v. 31, n. 6, pp. 1166–1180, 2018.

[15] AL-SHAROA, E., AL-KHASSAWENEH, M. A., AVIYENTE, S. "Detecting and tracking community structure in temporal networks: A low-rank+ sparse estimation based evolutionary clustering approach", *IEEE Transactions on Signal and Information Processing over Networks*, v. 5, n. 4, pp. 723–738, 2019.

[16] PAN, Y., HU, G., PAN, Z., et al. "An evolutionary approach based on symmetric nonnegative matrix factorization for community detection in dynamic networks", *IEICE TRANSACTIONS on Information and Systems*, v. 102, n. 12, pp. 2619–2623, 2019.

[17] FANI, H., JIANG, E., BAGHERI, E., et al. "User community detection via embedding of social network structure and temporal content", *Information Processing & Management*, v. 57, n. 2, pp. 102056, 2020.

[18] CHAN, S.-Y., HUI, P., XU, K. "Community detection of time-varying mobile social networks". In: *Complex Sciences: First International Conference, Complex 2009, Shanghai, China, February 23-25, 2009. Revised Papers, Part 1 1*, pp. 1154–1159, Berlin, 2009. Springer.

[19] WANG, P., FU, Y., ZHANG, J., et al. "Learning urban community structures: A collective embedding perspective with periodic spatial-temporal mobility graphs", *ACM Transactions on Intelligent Systems and Technology (TIST)*, v. 9, n. 6, pp. 1–28, 2018.

[20] FU, Y., LIU, G., GE, Y., et al. "Representing urban forms: A collective learning model with heterogeneous human mobility data", *IEEE transactions on knowledge and data engineering*, v. 31, n. 3, pp. 535–548, 2018.

[21] NIE, W.-P., ZHAO, Z.-D., CAI, S.-M., et al. "Understanding the urban mobility community by taxi travel trajectory", *Communications in Nonlinear Science and Numerical Simulation*, v. 101, pp. 105863, 2021.

[22] FERREIRA, D. L., NUNES, B. A., CAMPOS, C. A. V., et al. "A deep learning approach for identifying user communities based on geographical preferences and its applications to urban and environmental planning", *ACM Transactions on Spatial Algorithms and Systems (TSAS)*, v. 6, n. 3, pp. 1–24, 2020.

[23] HSIEH, K. *Machine learning systems for highly-distributed and rapidly-growing data*. Tese de Doutorado, Carnegie Mellon University, 2019.

[24] MCMAHAN, B., RAMAGE, D. "Federated learning: Collaborative machine learning without centralized training data". Availible at `https://ai.googleblog.com/2017/04/federated-learning-collaborative.html`, Apr 2017. Accessed: April 7, 2023.

[25] ALEDHARI, M., RAZZAK, R., PARIZI, R. M., et al. "Federated learning: A survey on enabling technologies, protocols, and applications", *IEEE Access*, v. 8, pp. 140699–140725, 2020.

[26] MCMAHAN, B., THAKURTA, A. "Federated Learning with Formal Differential Privacy Guarantees". Availible at `https://ai.googleblog.com/2022/02/federated-learning-with-formal.html`, Feb 2022. Accessed: April 2, 2023.

[27] FORTUNATO, S. "Community detection in graphs", *Physics reports*, v. 486, n. 3-5, pp. 75–174, 2010.

[28] ROSSETTI, G., CAZABET, R. "Community discovery in dynamic networks: a survey", *ACM computing surveys (CSUR)*, v. 51, n. 2, pp. 1–37, 2018.

[29] WU, C., SMITH, D., WANG, M. "Simulating the urban spatial structure with spatial interaction: A case study of urban polycentricity under different scenarios", *Computers, environment and urban systems*, v. 89, pp. 101677, 2021.

[30] JIANG, S., FERREIRA, J., GONZÁLEZ, M. C. "Clustering daily patterns of human activities in the city", *Data Mining and Knowledge Discovery*, v. 25, pp. 478–510, 2012.

[31] GREEN, N. "Functional polycentricity: A formal definition in terms of social network analysis", *Urban studies*, v. 44, n. 11, pp. 2077–2103, 2007.

[32] KIM, K. "Identifying the structure of cities by clustering using a new similarity measure based on smart card data", *IEEE Transactions on Intelligent Transportation Systems*, v. 21, n. 5, pp. 2002–2011, 2019.

[33] HE, M., GLASSER, J., BHAMIDI, S., et al. "Intertemporal community detection in human mobility networks". Availible at `https://doi.org/10.48550/arXiv.1906.04582`, 2019. Accessed: Jan 5, 2023.

[34] FIORINI, S., CIAVOTTA, M., MAURINO, A. "A multi-criteria algorithm for automatic detection of city communities". In: *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pp. 1266–1271, New York, 2021. IEEE.

[35] LIU, C., CHEN, L., YUAN, Q., et al. "Revealing dynamic spatial structures of urban mobility networks and the underlying evolutionary patterns", *ISPRS International Journal of Geo-Information*, v. 11, n. 4, pp. 237, 2022.

[36] ÇOLAK, S., LIMA, A., GONZÁLEZ, M. C. "Understanding congested travel in urban areas", *Nature communications*, v. 7, n. 1, pp. 10793, 2016.

[37] MENEGUETTE, R. I., DE GRANDE, R., LOUREIRO, A. "Intelligent transport system in smart cities", *Cham: Springer International Publishing*, v. 1, pp. 182, 2018.

[38] BARBOSA, H., BARTHELEMY, M., GHOSHAL, G., et al. "Human mobility: Models and applications", *Physics Reports*, v. 734, pp. 1–74, 2018.

[39] DEVI MURUGAVEL, K., RAMADASS, P., MAHENDRAN, R. K., et al. "Maintaining Effective Node Chain Connectivity in the Network with

Transmission Power of Self-Arranged AdHoc Routing in Cluster Scenario", *Electronics*, v. 11, n. 15, pp. 2455, 2022.

[40] YANG, L., CAO, X., HE, D., et al. "Modularity based community detection with deep learning." In: *IJCAI*, v. 16, pp. 2252–2258, 2016.

[41] JIN, D., LIU, Z., LI, W., et al. "Graph convolutional networks meet markov random fields: Semi-supervised community detection in attribute networks". In: *Proceedings of the AAAI conference on artificial intelligence*, v. 33, pp. 152–159, 2019.

[42] SU, X., XUE, S., LIU, F., et al. "A comprehensive survey on community detection with deep learning", *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[43] SOBOLEVSKY, S., BELYI, A. "Graph neural network inspired algorithm for unsupervised network community detection", *Applied Network Science*, v. 7, n. 1, pp. 63, 2022.

[44] CHEN, Z., LI, X., BRUNA, J. "Supervised community detection with line graph neural networks", *arXiv preprint arXiv:1705.08415*, 2017.

[45] WANG, X., LI, J., YANG, L., et al. "Unsupervised learning for community detection in attributed networks based on graph convolutional network", *Neurocomputing*, v. 456, pp. 147–155, 2021.

[46] SPERLÍ, G. "A deep learning based community detection approach". In: *Proceedings of the 34th ACM/SIGAPP symposium on applied computing*, pp. 1107–1110, 2019.

[47] DE SANTO, A., GALLI, A., MOSCATO, V., et al. "A deep learning approach for semi-supervised community detection in Online Social Networks", *Knowledge-Based Systems*, v. 229, pp. 107345, 2021.

[48] CAI, B., WANG, Y., ZENG, L., et al. "Edge classification based on convolutional neural networks for community detection in complex network", *Physica A: statistical mechanics and its applications*, v. 556, pp. 124826, 2020.

[49] JING, B., PARK, C., TONG, H. "Hdmi: High-order deep multiplex infomax". In: *Proceedings of the Web Conference 2021*, pp. 2414–2424, 2021.

[50] SOBOLEVSKY, S. "Deep Learning Attention Model for Supervised and Unsupervised Network Community Detection". In: *International Conference on Computational Science*, pp. 647–654. Springer, 2023.

[51] WANG, Z., WANG, C., GAO, C., et al. "An evolutionary autoencoder for dynamic community detection", *Science China Information Sciences*, v. 63, n. 11, pp. 212205, 2020.

[52] MASHHADI, A., STERNER, J., MURRAY, J. "Deep embedded clustering of urban communities using federated learning". In: *2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, New York, 2021. IEEE.

[53] TU, C., ZENG, X., WANG, H., et al. "A unified framework for community detection and network representation learning", *IEEE Transactions on Knowledge and Data Engineering*, v. 31, n. 6, pp. 1051–1065, 2018.

[54] GURUKAR, S., BOETTNER, B., BROWNING, C., et al. "Leveraging network representation learning and community detection for analyzing the activity profiles of adolescents", *Applied network science*, v. 7, n. 1, pp. 27, 2022.

[55] FERREIRA, D. L., NUNES, B. A. A., CAMPOS, C. A. V., et al. "User community identification through fine-grained mobility records for smart city applications", *IEEE Transactions on Intelligent Transportation Systems*, v. 23, n. 5, pp. 4387–4401, 2020.

[56] BANK, D., KOENIGSTEIN, N., GIRYES, R. "Autoencoders". `https://doi.org/10.48550/arXiv.2003.05991`, abr. 2009.

[57] BALDI, P. "Autoencoders, unsupervised learning, and deep architectures". In: *Proceedings of ICML workshop on unsupervised and transfer learning*, pp. 37–49. JMLR Workshop and Conference Proceedings, 2012.

[58] ROJAS, R. "The backpropagation algorithm". In: *Neural networks*, Springer, pp. 149–182, Berlin, 1996.

[59] YE, F., CHEN, C., ZHENG, Z. "Deep autoencoder-like nonnegative matrix factorization for community detection". In: *Proceedings of the 27th ACM international conference on information and knowledge management*, pp. 1393–1402, New, York, 2018. ACM.

[60] BHATIA, V., RANI, R. "A distributed overlapping community detection model for large graphs using autoencoder", *Future Generation Computer Systems*, v. 94, pp. 16–26, 2019.

[61] HE, C., ZHENG, Y., CHENG, J., et al. "Semi-supervised overlapping community detection in attributed graph with graph convolutional autoencoder", *Information Sciences*, v. 608, pp. 1464–1479, 2022.

[62] MASCI, J., MEIER, U., CIREŞAN, D., et al. "Stacked convolutional auto-encoders for hierarchical feature extraction". In: *Artificial Neural Networks and Machine Learning–ICANN 2011: 21st International Conference on Artificial Neural Networks, Espoo, Finland, June 14-17, 2011, Proceedings, Part I 21*, pp. 52–59, Berlin, 2011. Springer.

[63] KIM, P., KIM, P. "Convolutional neural network". In: MATLAB Deep Learning. Apress, Berkeley, CA. https://doi.org/10.1007/978-1-4842-2845-6_6, 2017.

[64] DUMOULIN, V., VISIN, F. "A guide to convolution arithmetic for deep learning". https://doi.org/10.48550/arXiv.1603.07285, jan. 2018.

[65] SHI, W., CABALLERO, J., THEIS, L., et al. "Is the deconvolution layer the same as a convolutional layer?" *arXiv preprint arXiv:1609.07009*, 2016.

[66] FENG, J., RONG, C., SUN, F., et al. "PMF: A privacy-preserving human mobility prediction framework via federated learning", *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, v. 4, n. 1, pp. 1–21, 2020.

[67] LI, A., WANG, S., LI, W., et al. "Predicting human mobility with federated learning". In: *Proceedings of the 28th International Conference on Advances in Geographic Information Systems*, pp. 441–444, New York, NY, USA, 2020. ACM.

[68] XIA, M., JIN, D., CHEN, J. "Short-term traffic flow prediction based on graph convolutional networks and federated learning", *IEEE Transactions on Intelligent Transportation Systems*, v. 24, pp. 1191 – 1203, 2022.

[69] KONEČNỲ, J., MCMAHAN, H. B., RAMAGE, D., et al. "Federated optimization: Distributed machine learning for on-device intelligence". Availible at https://doi.org/10.48550/arXiv.1610.02527, 2016. Accessed: Jan 9, 2023.

[70] MCMAHAN, B., MOORE, E., RAMAGE, D., et al. "Communication-efficient learning of deep networks from decentralized data". In: *Artificial intelligence and statistics*, pp. 1273–1282, Lauderdale, 2017. PMLR.

[71] LI, T., SAHU, A. K., TALWALKAR, A., et al. "Federated learning: Challenges, methods, and future directions", *IEEE Signal Processing Magazine*, v. 37, n. 3, pp. 50–60, 2020.

[72] MCMAHAN, H. B., MOORE, E., RAMAGE, D., et al. "Federated learning of deep networks using model averaging", *arXiv preprint arXiv:1602.05629*, v. 2, 2016.

[73] YANG, Q., LIU, Y., CHEN, T., et al. "Federated machine learning: Concept and applications", *ACM Transactions on Intelligent Systems and Technology (TIST)*, v. 10, n. 2, pp. 1–19, 2019.

[74] YANG, Q., LIU, Y., CHENG, Y., et al. "Federated learning", *Synthesis Lectures on Artificial Intelligence and Machine Learning*, v. 13, n. 3, pp. 1–207, 2019.

[75] BONAWITZ, K., IVANOV, V., KREUTER, B., et al. "Practical secure aggregation for privacy-preserving machine learning". In: *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1175–1191, New, York, 2017. ACM.

[76] AONO, Y., HAYASHI, T., WANG, L., et al. "Privacy-preserving deep learning via additively homomorphic encryption", *IEEE Transactions on Information Forensics and Security*, v. 13, n. 5, pp. 1333–1345, 2017.

[77] LIU, Y., CHEN, T., YANG, Q. "Secure federated transfer learning. CoRR abs/1812.03337 (2018)", *arXiv preprint arXiv:1812.03337*, 2018.

[78] DWORK, C. "Differential privacy: A survey of results". In: *International conference on theory and applications of models of computation*, pp. 1–19, Berlin, Heidelberg, 2008. Springer.

[79] GEYER, R. C., KLEIN, T., NABI, M. "Differentially private federated learning: A client level perspective", *arXiv preprint arXiv:1712.07557*, 2017.

[80] ABADI, M., AGARWAL, A., BARHAM, P., et al. "TensorFlow: Large-scale machine learning on heterogeneous systems". 2015.

[81] LI, X., HUANG, K., YANG, W., et al. "On the convergence of fedavg on non-iid data", *arXiv preprint arXiv:1907.02189*, 2019.

[82] XIE, C., KOYEJO, S., GUPTA, I. "Asynchronous federated optimization", *arXiv preprint arXiv:1903.03934*, 2019.

[83] DAI, W., ZHOU, Y., DONG, N., et al. "Toward understanding the impact of staleness in distributed machine learning", *arXiv preprint arXiv:1810.03264*, 2018.

[84] LI, T., SAHU, A. K., ZAHEER, M., et al. "Federated optimization in heterogeneous networks", *Proceedings of Machine learning and systems*, v. 2, pp. 429–450, 2020.

[85] NARDI, M., VALERIO, L., PASSARELLA, A. "Anomaly detection through unsupervised federated learning". In: *2022 18th International Conference on Mobility, Sensing and Networking (MSN)*, pp. 495–501, New York, 2022. IEEE.

[86] AHMED, U., SRIVASTAVA, G., LIN, J. C.-W. "Reliable customer analysis using federated learning and exploring deep-attention edge intelligence", *Future Generation Computer Systems*, v. 127, pp. 70–79, 2022.

[87] DASGUPTA, A., RAFTERY, A. E. "Detecting features in spatial point processes with clutter via model-based clustering", *Journal of the American statistical Association*, v. 93, n. 441, pp. 294–302, 1998.

[88] JOHNSON, S. C. "Hierarchical clustering schemes", *Psychometrika*, v. 32, n. 3, pp. 241–254, 1967.

[89] LIKAS, A., VLASSIS, N., VERBEEK, J. J. "The global k-means clustering algorithm", *Pattern recognition*, v. 36, n. 2, pp. 451–461, 2003.

[90] BEZDEK, J. C., EHRLICH, R., FULL, W. "FCM: The fuzzy c-means clustering algorithm", *Computers & geosciences*, v. 10, n. 2-3, pp. 191–203, 1984.

[91] REYNOLDS, D. A. "Gaussian mixture models", *Encyclopedia of biometrics*, v. 741, n. 659-663, pp. 827–832, 2009.

[92] AKAIKE, H. "Information theory and an extension of the maximum likelihood principle". In: *Selected papers of hirotugu akaike*, Springer, pp. 199–213, New York, 1998.

[93] SCHWARZ, G. "Estimating the dimension of a model", *The annals of statistics*, pp. 461–464, 1978.

[94] MURPHY, K. P. *Machine learning: a probabilistic perspective*. USA, MIT press, 2012.

[95] HASTIE, T., TIBSHIRANI, R., FRIEDMAN, J. *The elements of statistical learning: data mining, inference, and prediction. Springer series in statistics*. New York, Springer Series in Statistics, 2009.

[96] ZEBARI, R., ABDULAZEEZ, A., ZEEBAREE, D., et al. "A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction", *J. Appl. Sci. Technol. Trends*, v. 1, n. 2, pp. 56–70, 2020.

[97] JAEGER, T. F. "Categorical data analysis: Away from ANOVAs (transformation or not) and towards logit mixed models", *Journal of memory and language*, v. 59, n. 4, pp. 434–446, 2008.

[98] ALBAWI, S., MOHAMMED, T. A., AL-ZAWI, S. "Understanding of a convolutional neural network". In: *2017 international conference on engineering and technology (ICET)*, pp. 1–6, New York, 2017. IEEE.

[99] KAIROUZ, P., MCMAHAN, H. B., AVENT, B., et al. "Advances and open problems in federated learning", *Foundations and Trends® in Machine Learning*, v. 14, n. 1–2, pp. 1–210, 2021.

[100] NASR, M., SHOKRI, R., HOUMANSADR, A. "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning". In: *2019 IEEE symposium on security and privacy (SP)*, pp. 739–753, San Francisco, 2019. IEEE.

[101] MELIS, L., SONG, C., DE CRISTOFARO, E., et al. "Exploiting unintended feature leakage in collaborative learning". In: *2019 IEEE symposium on security and privacy (SP)*, pp. 691–706, New York, 2019. IEEE.

[102] CELEBI, M. E., AYDIN, K. *Unsupervised learning algorithms*. USA, Springer, 2016.

[103] KONISHI, S., KITAGAWA, G. "Bayesian information criteria", *Information criteria and statistical modeling*, pp. 211–237, 2008.

[104] PIORKOWSKI, M., SARAFIJANOVIC-DJUKIC, N., GROSSGLAUSER, M. "CRAWDAD dataset epfl/mobility (v. 2009-02-24)". Downloaded from `https://crawdad.org/epfl/mobility/20090224/cab`, fev. 2009. traceset: cab.

[105] BRACCIALE, L., BONOLA, M., LORETI, P., et al. "CRAWDAD dataset roma/taxi (v. 2014-07-17)". Downloaded from `https://crawdad.org/roma/taxi/20140717`, jul. 2014.

[106] DOT, U. "Next generation simulation (NGSIM) vehicle trajectories and supporting data", *US Department of Transportation*, v. 1, 2018.

[107] WANG, Z., BOVIK, A. C., SHEIKH, H. R., et al. "Image quality assessment: from error visibility to structural similarity", *IEEE transactions on image processing*, v. 13, n. 4, pp. 600–612, 2004.

[108] HUBERT, L., ARABIE, P. "Comparing partitions", *Journal of classification*, v. 2, n. 1, pp. 193–218, 1985.

[109] SANTOS, J. M., EMBRECHTS, M. "On the use of the adjusted rand index as a metric for evaluating supervised classification". In: *International conference on artificial neural networks*, pp. 175–184, Berlin, 2009. Springer.

[110] ABDULRAHMAN, S., TOUT, H., OULD-SLIMANE, H., et al. "A survey on federated learning: The journey from centralized to distributed on-site learning and beyond", *IEEE Internet of Things Journal*, v. 8, n. 7, pp. 5476–5497, 2020.

[111] WEI, K., GUO, S., ZENG, D., et al. "Exploiting small world properties for message forwarding in delay tolerant networks", *IEEE transactions on computers*, v. 64, n. 10, pp. 2809–2818, 2015.

[112] SOBIN, C., RAYCHOUDHURY, V., MARFIA, G., et al. "A survey of routing and data dissemination in delay tolerant networks", *Journal of Network and Computer Applications*, v. 67, pp. 128–146, 2016.

[113] WEI, K., ZENG, D., GUO, S., et al. "On social delay-tolerant networking: Aggregation, tie detection, and routing", *IEEE Transactions on Parallel and Distributed Systems*, v. 25, n. 6, pp. 1563–1573, 2013.

[114] ZHANG, J., HUANG, H., MIN, G., et al. "Social-aware routing in mobile opportunistic networks", *IEEE Wireless Communications*, v. 28, n. 2, pp. 152–158, 2021.

[115] XIA, F., LIU, L., LI, J., et al. "Socially aware networking: A survey", *IEEE Systems Journal*, v. 9, n. 3, pp. 904–921, 2013.

[116] CAI, Y., ZHANG, H., FAN, Y., et al. "A survey on routing algorithms for opportunistic mobile social networks", *China Communications*, v. 18, n. 2, pp. 86–109, 2021.

[117] ZHU, Y., XU, B., SHI, X., et al. "A survey of social-based routing in delay tolerant networks: Positive and negative social effects", *IEEE Communications Surveys & Tutorials*, v. 15, n. 1, pp. 387–401, 2012.

[118] WEI, K., LIANG, X., XU, K. "A survey of social-aware routing protocols in delay tolerant networks: Applications, taxonomy and design-related issues", *IEEE Communications Surveys & Tutorials*, v. 16, n. 1, pp. 556–578, 2013.

[119] DALY, E. M., HAAHR, M. "Social network analysis for routing in disconnected delay-tolerant manets". In: *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*, pp. 32–40, 2007.

[120] HUI, P., CROWCROFT, J., YONEKI, E. "Bubble rap: social-based forwarding in delay tolerant networks". In: *Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, pp. 241–250, 2008.

[121] LENANDO, H., ALRFAAY, M. "EpSoc: Social-based epidemic-based routing protocol in opportunistic mobile social network", *Mobile Information Systems*, v. 2018, n. 1, pp. 6462826, 2018.

[122] KERÄNEN, A., OTT, J., KÄRKKÄINEN, T. "The ONE Simulator for DTN Protocol Evaluation". In: *SIMUTools '09: Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, New York, NY, USA, 2009. ICST. ISBN: 978-963-9799-45-5.

[123] VAHDAT, A. "Epidemic Routing for Partially Connected Ad Hoc Networks", *Technical Report CS-200006, Duke University, Durham, April*, 2000.

[124] TELANG, S., CHEL, A., NEMADE, A., et al. "Intelligent transport system for a smart city", *Security and privacy applications for smart city development*, v. 308, pp. 171–187, 2021.

[125] YANG, Y., BAGRODIA, R. "Evaluation of VANET-based advanced intelligent transportation systems". In: *Proceedings of the sixth ACM international workshop on VehiculAr InterNETworking*, pp. 3–12, 2009.